



5-2003

Markov chain testing models for "sequential-stage system reliability growth via failure mode removal"

Michael W. Corum

Recommended Citation

Corum, Michael W., "Markov chain testing models for "sequential-stage system reliability growth via failure mode removal". " Master's Thesis, University of Tennessee, 2003.
https://trace.tennessee.edu/utk_gradthes/5203

This Thesis is brought to you for free and open access by the Graduate School at Trace: Tennessee Research and Creative Exchange. It has been accepted for inclusion in Masters Theses by an authorized administrator of Trace: Tennessee Research and Creative Exchange. For more information, please contact trace@utk.edu.

To the Graduate Council:

I am submitting herewith a thesis written by Michael W. Corum entitled "Markov chain testing models for "sequential-stage system reliability growth via failure mode removal"." I have examined the final electronic copy of this thesis for form and content and recommend that it be accepted in partial fulfillment of the requirements for the degree of Master of Science, with a major in Computer Science.

Jesse Poore, Major Professor

We have read this thesis and recommend its acceptance:

Accepted for the Council:


Dixie L. Thompson

Vice Provost and Dean of the Graduate School

(Original signatures are on file with official student records.)

To the Graduate Council:

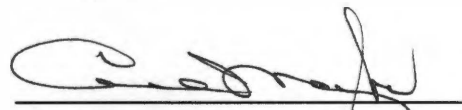
I am submitting herewith a thesis written by Michael W. Corum entitled "Markov Chain Testing Models For 'Sequential-Stage System Reliability Growth Via Failure Mode Removal'." I have examined the final paper copy of this thesis for form and content and recommend that it be accepted in partial fulfillment of the requirements for the degree of Master of Science, with a major in Computer Science.


Jesse Poore, Major Professor

We have read this thesis
and recommend its acceptance:


Michael Thomason
Stacy Provell

Acceptance for the council:


Vice Provost and Dean of Graduate Studies

MARKOV CHAIN TESTING MODELS FOR “SEQUENTIAL- STAGE SYSTEM RELIABILITY GROWTH *VIA* FAILURE MODE REMOVAL”

A Thesis

Presented for the

Master of Science Degree

The University of Tennessee, Knoxville

Michael W. Corum

May 2003

Thesis

2003

.C679

Copyright © Michael W. Corum, 2003

All rights reserved.

Dedication

This thesis is dedicated to my wife, Susan Corum, for her support,
confidence, and encouragement to achieve my goals.

Acknowledgments

I wish to thank all of those who helped me in completing Master of Science in Computer Science. I thank Dr. Jesse Poore for all the hard work and guidance he has given me with this thesis. I would also like to thank Dr. Stacy Prowell for his help in solving many problems that arose during the research, and Dr. Michael Thomason for his careful review of this thesis. Also, I would like to thank the members of the SQRL lab for their assistance in this research. Finally, I wish to express thanks to D. Gaver, P. Jacobs, K. Glazebrook, and E. Seglie for interesting questions and exemplary mathematical analysis.

Abstract

The goal of this research was to recast the issues raised in “Probability Models for Sequential-Stage System Reliability Growth via Failure Mode Removal” by Gaver, et. al. [1] into Markov chain models as treated by the Software Quality Research Laboratory (SQRL). Solutions given by Gaver were studied and Markov chain testing models were proposed as alternative solutions, providing more questions and answers that are relevant to the testing of sequential-stage systems. Reformulation of the questions yielded solutions in the form of familiar statistics of the Markov chain.

Contents

CHAPTER 1: Introduction.....	1
1.1 Background.....	1
1.2 Paraphrase of SRG	2
1.3 Notation.....	5
1.4 Motivation for the Alternative Approach	6
CHAPTER 2: The Testing Model	8
2.1 Definition of the Testing Model	8
2.2 Assigning the Probability Mass	11
2.3 Derivation of Analytical Results.....	11
2.3.1 Probabilities of Occurrence and Long Run Occupancies of States	12
2.3.2 Derivation of the Mean First Passage of the Model Sink.....	13
CHAPTER 3: Question 1 - Determine Field Reliability	17
3.1 Computing the Field Reliability	17
3.1.1 Stage-Wise Survival Probabilities of the Fielded System	18
3.1.2 Joint Probability of Defects Remaining in Each Stage After t Tests	19
3.1.3 Probability of System Survival in the Field After t Tests.....	20
3.2 Larger Question and Answer Space	21
3.2.1 Further Questions Concerning the Testing Model	22
3.2.2 Further Questions Concerning the States of the Testing Model.....	22
3.2.3 Further Questions Concerning the Stimuli of the Testing Model.....	23
3.2.4 Further Questions Concerning the Arcs of the Testing Model.....	24
3.3 An Example of System Reliability Growth	25
CHAPTER 4: Question 2 - Determine the jth Success.....	29
4.1 The j -Success Model.....	29

4.2	Computing the Expected Value of the j th Success	35
4.2.1	Computing the Long Run Occupancies	35
4.2.2	Computing the Mean First Passage of the Model Sink.	36
4.3	Larger Question and Answer Space	37
4.3.1	Further Questions Concerning the j -Success Testing Model	38
4.3.2	Further Questions Concerning the Stimuli of the j -Success Model	38
4.4	An Example of Computing the j th Success	39
CHAPTER 5: Question 3 - The rth Consecutive Successes.....		44
5.1	Consecutive Success Model	44
5.2	The Expected Value of the r th Consecutive Success	48
5.2.1	Computing Probabilities of Occurrence and Long Run Occupancies ...	48
5.2.2	Computing the Mean First Passage of the Model Sink.	49
5.3	Larger Question and Answer Space	49
5.3.1	Further Questions About States of the Consecutive Success Model	50
5.3.2	Further Questions About Stimuli of the Consecutive Success Model ...	50
5.4	An Example of the r th Consecutive Success	51
CHAPTER 6: Question 4 - Determine Failure Characteristics		56
CHAPTER 7: Summary and Conclusion		58
References.....		60
Appendix.....		62
A.1	Algorithm 1 - The Single-Step Transition Matrix	63
A.2	Algorithm 2 - Expected Reliability After t Tests	64
A.3	Algorithm 3 - Expected Value of the j th Success.....	65
A.4	Algorithm 4 - Mean of the r th Consecutive Success	68
Vita		71

Tables

Table 3.1: Defect Survival Probabilities for Example System.	25
Table 3.2: Stimulus Statistics for Example System.	28
Table 3.3: Statistics for Incoming Arcs of State test_0_0_0_0.	28
Table 4.1: Expected Number of Tests Required to Achieve j Successes.	40
Table 4.2: Additional Results Concerning the Stimuli.	43
Table 5.1: Ordering of Sates in the Consecutive Success Model.	47
Table 5.2: Expected Number of Tests Required to Achieve r Consecutive Successes.	52
Table 5.3: Success and Failure Data for $r = 3$	55
Table 5.4: Mean Number of Tests Applied with Given Consecutive Successes.	55
Table 6.1: Probability of Mission Success.	57

Figures

Figure 2.1: Two-Stage Testing Model with Three Defects Per Stage.....	10
Figure 3.1: Probability of System Survival After t Tests.....	26
Figure 4.1: Example j -Success Model.....	31
Figure 4.2: Tiered Structure of j -Success Model.....	32
Figure 4.3: 3-Success Model for Two-Stage System with 3 Defects Per Stage.....	34
Figure 4.4: Probability of System Survival After j Successes.....	41
Figure 5.1: Example State Transition Diagram of the Consecutive Success Model....	46
Figure 5.2: Probability of System Success after r Consecutive Successes.	53

Chapter 1:

Introduction

1.1 Background

The paper entitled “Probability Models for Sequential-Stage System Reliability Growth via Failure Mode Removal” by Gaver, et.al. [1] is the basis for this research. Because we refer to this paper often, we will abbreviate the work as “SRG” for system reliability growth or refer to it as “Gaver.” In SRG we are asked to consider a sequential-stage software system in which an unknown number of design defects, or failure modes, are initially present in each stage. Further, suppose that the number of defects is reduced stochastically at each test. Questions addressing the reliability of such a system after testing are posed. Mathematical models are presented which describe the effect of end-to-end or linked-stage testing, and defect identification and removal, on fielded system reliability. Gaver noted that the process under study is Markovian and used that fact in various analyses, but we will show that this property can be further exploited.

The Software Quality Research Laboratory (SQRL) at the University of Tennessee has worked in the area of encoding usage models of systems into Markov chains for purposes of testing and reliability assessment for several years [2]. In this thesis research we recast

the issues of SRG into Markov chain models as treated by SQLR. We will show that this alternative treatment results in a better understanding of the original question and answer space, as well as suggesting additional questions (and answers) that are relevant to testing sequential stage systems. Specifically, we will construct Markov chain testing models to answer questions posed in SRG, as well as other relevant questions useful to a test planner.

1.2 Paraphrase of SRG

The first step is to paraphrase the problem as presented by Gaver. The indented material is not a direct quotation in total, however much of it is taken word for word from Gaver.

The system to be tested, denoted S , is composed of a number, $s > 0$, of operational stages, with each stage containing some number of design defects, or failure modes. Execution of the software is sequential, moving from stage to stage, in which each stage must execute on demand for successful operation. Success is defined as an execution in which no design defect is activated upon execution of Stage i , and thus a demand is placed upon Stage $i + 1$. Thus, stages are request-activated strictly serially, starting with Stage 1 and ending with Stage s .

Because of the sequential nature of the system under test, failure is defined as the activation of one or more design defects in a stage when demand is placed on that stage. The defect could be contained properly within the stage, or it

may be triggered as part of the interaction between Stages i and $i+1$. In the later case the defect shall be assumed to “belong” to Stage i for testing purposes, although interaction between stages may also be viewed as a stage. Thus, failure of any stage means total system failure, and successful operation of the system implies end-to-end functionality.

There are several testing conditions and simplifying assumptions which are applicable to the above system. Initially, each stage, i , contains some number of design defects, denoted $D_i(0)$. It is assumed that this number of defects is random, or at least unknown. Each test of the system consists of sequentially testing each stage, with the possible outcomes of failure, i.e., defect activation and identification, or success, in which no design defects are activated.

A test activates a particular design defect in Stage i with probability $\bar{\theta}_i$. A defect is not activated, or survives, with probability $1 - \bar{\theta}_i = \theta_i$. These probabilities are assumed to be independent between stages, and fixed from test to test. Thus, the test of each stage is a Bernoulli trial, and the test of multiple stages is a Bernoulli process.

We next define the stage-wise defect activation and survival probabilities. Let $D_i(0)$ denote the number of defects initially present in Stage i , $i = 1, \dots, s$. Using binomial probabilities, define the probability that some defect in Stage i becomes active as $\tilde{p}_i(d_i)$. Thus, the probability that a defect in Stage i remains

inactive, or survives, is $\tilde{q}_i(d_i) = 1 - \tilde{p}_i(d_i)$. Since each defect in a stage survives with probability θ_i , and there are d_i defects present in Stage i , the stage-wise survival probabilities are $\tilde{q}_i(d_i) = \theta_i^{d_i}$, and $\tilde{p}_i(d_i) = 1 - \theta_i^{d_i}$. The stage-wise survival probabilities will change as defects are identified and removed, but θ_i remains constant throughout testing.

It is assumed that at most one design defect is identified and removed per test. If more than one defect becomes active upon test of a stage, then only one defect is identified and removed; the others may activate again. An activated defect is assumed to be removed from the system without introducing additional errors into the system.

Gaver provides solutions to the following questions:

1. After a given number of system tests, what is the (approximate) probability that the system will operate satisfactorily (not fail) when released to the field or delivered to a user?
2. How many tests are likely to be required to achieve the first (or j th) end-to-end success?
3. How many tests are required to achieve r (e.g. 3 or 5) consecutive end-to-end test successes, or, in statistical parlance, a (first) run of r ?

4. Suppose testing is stopped after T tests, after which no further design modifications are contemplated. What are the failure characteristics of the system if fielded: e.g. what is the operational/field probability of system (reliability) success? What is the probability that the system completes a mission that requires M successes if $M+R$ systems are allocated? What is the mean, and variability, of the number of tests required?

1.3 Notation

Notation	Description
S	The sequential-stage system under test, $s > 0$
s	Number of system stages
d_i	Number of design defects (failure modes) for Stage i
$\bar{\theta}_i$	Probability that a defect in Stage i becomes active, $0 \leq \bar{\theta}_i \leq 1$
θ_i	Probability that a defect in Stage i does not become active, $\theta_i = 1 - \bar{\theta}_i$
$D_i(t)$	Number of design defects present in Stage i after t tests, $D_i(0)$ design defects present initially in Stage i
$\tilde{p}_i(d_i)$	Stage-wise activation (failure) probabilities, $\tilde{p}_i(d_i) = 1 - \theta_i^{d_i}$
$\tilde{q}_i(d_i)$	Stage-wise survival probabilities, $\tilde{q}_i(d_i) = \theta_i^{d_i}$

Notation	Description
Q_i	Probability that a remaining defect in Stage i does not activate while the system is put in use in the field during one mission, desirably, $Q_i \approx$ or $> \tilde{q}_i(d_i)$.
$Q(i)$	Probability of system survival in the field, where i represents the state of the model with the specified distribution of defects.
$p(d_1, \dots, d_s, t)$	Joint probability of the number of defects present in each stage after t tests
$\tilde{Q}(t)$	Probability of system survival in the field after t tests
$r_r(d_1, \dots, d_s)$	Conditional expected time (number of tests) until a run of r successes first occurs, given there are initially d_i defects in Stage i
n	The number of states in the Markov chain testing model
P	$n \times n$ Single-step transition matrix for the Markov chain
M	Matrix of mean first passage times
V	Matrix of the variance of the mean first passage times
α	Vector of long run occupancies
$m_{i,j}^{(2)}$	The expectation of the square of the mean first passage from state i to state j .
y	Vector of probabilities of occurrence of the states in the testing process
ρ	Arbitrary path from state i to state j .
β	$n \times 1$ vector of probabilities of success

1.4 Motivation for the Alternative Approach

Markov chains are used to design system usage models representing the states of use of a software intensive system, yielding numerous statistical and analytical results for

model validation and system test planning. The Markov chain testing models that will be developed herein are similar to usage models, for which SQRL has many results formally derived, and software tools to support application.

The explicit treatment of SRG problems in Markov chains also provides a great deal of familiar mathematical structure. This structure may be used to gain a better understanding of the question and answer space of the problem, thereby providing further information about the testing process.

There is a single-step transition matrix, P , whose entries $p_{i,j}$ represent the probability of making a state transition to state j , given that the process is in state i , and the initial probability vector, π_0 , with all components, excepting the component corresponding to the unique model source, equal to zero. The component which corresponds to the model source is set equal to one, since the process may only be in the source before any tests have been applied. The initial probability vector and transition matrix completely determine the unique Markov chain [3].

Thus, additional operationally relevant questions beyond those posed in SRG may be obtained directly, or derived from standard analysis. These include results concerning the behavior of testing strategies in the long run, such as long run state occupancies and probabilities of occurrence of the states of the model in a realization, as well as useful information about the associated arcs and stimuli of the model. Markov chain analysis yields the variance associated with many of the expectations provided in answer to the questions in SRG.

Chapter 2:

The Testing Model

This chapter describes the structure of the Markov chain testing model designed to answer questions 1-4 of SRG and some basic analysis.

2.1 Definition of the Testing Model

Using the simplifying assumptions stated in the previous chapter, we construct the testing model as follows: the system to be tested consists of s stages, with $D_i(0)$ defects initially present in Stage i , with s , $D_i(0)$ non-negative integers and $1 \leq i \leq s$. Each state of the model represents a unique distribution of defects among each of the s stages. The states of the model may be denoted “test_ $d_1 \dots d_s$,” signifying that the state represents testing the system with $0 \leq d_i \leq D_i(0)$ defects in each stage. There are $D_i(0) + 1$ possible values of the number of defects within Stage i . Thus, there are $n = \prod_{i=1}^s (D_i(0) + 1)$ states in the model.

Each arc of the model represents a possible outcome of one test of the system. Each state of the model has one arc, labeled “success,” that is a loop representing a successful end-to-end test of the system. Each state has at most s failure arcs, one for each of the s stages, representing a failure in that stage, labeled “failed_ $stage_i$.” Note that when $d_i = 0$, there are no failures to activate, and thus there is no failure arc associated with this stage (lower right hand corner of Figure 2.1). Each state will have a maximum of $s + 1$ outgoing arcs, and hence there is a maximum of $(s + 1)n$ arcs in the model.

The target state of an arc representing a failure in Stage i is the state with one less defect in Stage i , and the same number of defects in all other stages. For arcs which represent a successful test of the system, the target state is the same as the source state, since no defects have been identified and removed. Since the testing process either stays in the current state or makes the transition to a state with one less defect, the structure of the single-step transition matrix, P , is upper triangular. The chain is then made recurrent to facilitate later computations. This is accomplished by setting $P_{n,1} = 1$, making it the only non-zero entry below the main diagonal.

Consider, for example, that the system under test is composed of two stages, with three defects initially present in each stage. In this instance we have a total of $4^2 = 16$ states in the model. Figure 2.1 depicts the state diagram of the example system. Note that the model source, indicated by the start arrow (upper left hand corner), is the state “test_3_3”, and that the model sink is the state “test_0_0.”

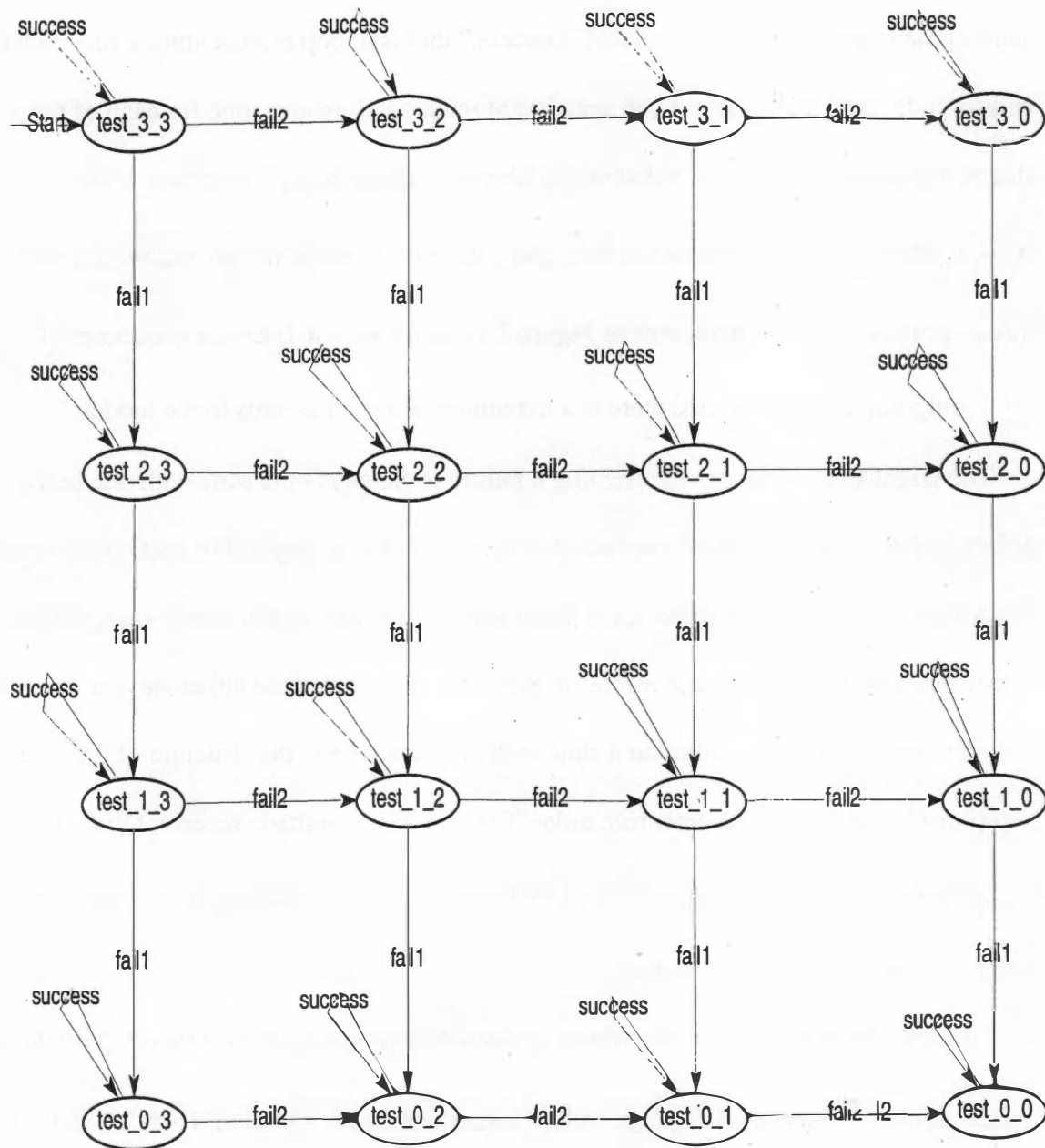


Figure 2.1: Two-Stage Testing Model with Three Defects Per Stage.

2.2 Assigning the Probability Mass

The probability mass is distributed across the arcs of the model as follows. Because the stage-wise failure probabilities are independent between stages, the probability of an end-to-end success of the system is given by

$$\theta_1^{d_1} \cdot \dots \cdot \theta_s^{d_s} = \tilde{q}_1(d_1) \cdot \dots \cdot \tilde{q}_s(d_s) = \prod_{i=1}^s \tilde{q}_i(d_i). \text{ Since there are no defects identified and}$$

removed on a successful test of the system, these probabilities are assigned to the entries on the main diagonal of the single-step transition matrix.

Because of the sequential nature of the system under test, a failure in Stage i may occur if and only if the tests of all previous stages were successful. Thus, the arc represent-

ing a failure in Stage i is assigned the probability $\left(\prod_{j=1}^{i-1} \tilde{q}_j(d_j) \right) (1 - \tilde{q}_i(d_i))$. The probabil-

ity of defect activation in Stage 1 is given by $1 - \tilde{q}_1(d_1) = \tilde{p}_1(d_1)$.

2.3 Derivation of Analytical Results

The regular structure of the Markov chain testing model permits simplification of the computation of several useful results. Because of the combinatorial growth in the number of states we use the regular structure to avoid the traditional computation by matrix inversions [3]. The near-upper triangular form of the single-step transition matrix allows the probability of occurrence and long run occupancy of each state to be computed directly.

This in turn allows for the direct computation of the mean first passage times of the model sink (one for each state), and their associated variances, as used later in Chapters 4 and 5.

2.3.1 Probabilities of Occurrence and Long Run Occupancies of States

One method of computing the vector of long-run occupancies is to normalize the components of the vector of the probability of occurrence of each state in the testing process. The normalized i th component of this vector then gives the long-run occupancy of state i . In general, there may be one or more disjoint sets of transient states within a given Markov chain. As such, the computation of the vector of probabilities of occurrence is complicated by the recurrence loops within the process. Solutions for the probability of occurrence of the states within such a model may be found in [3] and [4].

In chapters 4 and 5 we modify the structure of the proposed model to eliminate the recurrence loops within the chain, thus simplifying the computation. Redirecting the recurrent arcs from their sources to the model sink may be accomplished by a permutation of the single step transition matrix. This ensures that the process moves forward on every test; no state is visited more than once.

Application of this structure to the problem simplifies the computation of the probability of occurrence of a state as follows. Since the process can never return to a state once it has left that state, the probability of occurrence may be obtained by taking the sum over all paths leading to the state of the product of the probabilities assigned to the arcs along that path. More succinctly, letting y denote the vector of probabilities of occurrence, and letting ρ denote an arbitrary path of length $|\rho|$ from the model source to state i , we have

$y_i = \sum_{\rho} \left(\prod_{j=1}^{|\rho|-1} P_{\rho_j \rho_{j+1}} \right)$. The i th component of the vector of long run occupancies may

then be calculated as $\alpha_i = \frac{y_i}{\sum_{j=1}^n y_j}$, where $1 \leq i \leq n$.

Consider, for example, testing a four stage system with three defects initially present in each stage, and that the process is currently in state test_2_3_1_0. There are only three states that lead to the current state, namely, test_3_3_1_0, test_2_3_2_0, and state test_2_3_1_1. The probability of occurrence of state test_2_3_1_0 may be computed by taking the sum of the probability of occurrence for each of the three states multiplied by the probability of making the transition from said state to test_2_3_1_0.

2.3.2 Derivation of the Mean First Passage of the Model Sink

Given that the vector of long-run occupancies has been calculated, the mean first passage times of the model sink (one for each state in the model) may be calculated as follows. From [3], we know that the mean first passage of state j , given that the process began in state i , is given by $m_{i,j} = 1 + \sum_{k \neq j} p_{i,k} m_{k,j}$ when $i \neq j$, and $m_{i,i} = \frac{1}{\alpha_i}$. Solutions generally make use of the fundamental matrix presented in [3]. However, when $i \neq j$, $i < n$, the general solution requires a matrix inversion, and is therefore undesirable for sequential stage systems with many stages and defects per stage. The following theorem is presented

as a direct approach to calculate the mean first passage time of the model sink from state i ,

$$1 \leq i < n.$$

$$m_{i,n} = \frac{1 + \sum_{k=i+1}^{n-1} p_{i,k} m_{k,n}}{1 - p_{i,i}}$$

Theorem 2.1 , for $1 \leq i < n$, and $p_{i,i} \neq 1$.

Proof

$$m_{i,n} = 1 + \sum_{k \neq n} p_{i,k} m_{k,n} \quad (\text{By given})$$

$$m_{i,n} = 1 + \sum_{k=1}^{n-1} p_{i,k} m_{k,n} \quad (\text{Replace sum})$$

$$m_{i,n} = 1 + \sum_{k=i}^{n-1} p_{i,k} m_{k,n} \quad (j < i \Rightarrow p_{i,j} = 0, \quad i \neq n)$$

$$m_{i,n} = 1 + p_{i,i} m_{i,n} + \sum_{k=i+1}^{n-1} p_{i,k} m_{k,n} \quad (\text{Expand sum})$$

$$m_{i,n}(1 - p_{i,i}) = 1 + \sum_{k=i+1}^{n-1} p_{i,k} m_{k,n} \quad (\text{Factor, } m_{i,n} \neq 0)$$

$$m_{i,n} = \frac{1 + \sum_{k=i+1}^{n-1} p_{i,k} m_{k,n}}{1 - p_{i,i}} \quad (p_{i,i} \neq 1) \quad \square$$

The variance of a random variable X is given by $\text{Var}(X) = E(X^2) - E^2(X)$. A

solution for the variance of the mean first passage times is also given in [3], as

$v_{i,j} = \sum_{k \neq j} p_{i,k} m_{k,j}^{(2)} + 2 \sum_{k \neq j} p_{i,k} m_{k,j} + 1 - m_{i,j}^2$. The result of primary interest is the

mean first passage of the model sink given that the process started in the model source.

Making use of the regular structure of the model, the variance of the mean first passage times of the model sink (one for each state in the model) may also be computed directly.

$$\text{Theorem 2.2} \quad v_{i,n} = \frac{1 + \sum_{k=i+1}^{n-1} p_{i,k} m_{k,n}^{(2)} + 2 \sum_{k=i}^{n-1} p_{i,k} m_{k,n}}{1 - p_{i,i}} - m_{i,n}^2, \text{ for } 1 \leq i < n, \text{ and}$$

$$p_{i,i} \neq 1.$$

Proof

$$v_{i,n} = 1 + \sum_{k \neq n} p_{i,k} m_{k,n}^{(2)} + 2 \sum_{k \neq n} p_{i,k} m_{k,n} - m_{i,n}^2 \quad (\text{By given})$$

$$m_{i,n}^{(2)} - m_{i,n}^2 = 1 + \sum_{k \neq n} p_{i,k} m_{k,n}^{(2)} + 2 \sum_{k \neq n} p_{i,k} m_{k,n} - m_{i,n}^2 \quad (v_{i,n} = m_{i,n}^{(2)} - m_{i,n}^2)$$

$$m_{i,n}^{(2)} = 1 + \sum_{k \neq n} p_{i,k} m_{k,n}^{(2)} + 2 \sum_{k \neq n} p_{i,k} m_{k,n} \quad (\text{Add } m_{i,n}^2 \text{ to both sides})$$

$$m_{i,n}^{(2)} = 1 + \sum_{k=1}^{n-1} p_{i,k} m_{k,n}^{(2)} + 2 \sum_{k=1}^{n-1} p_{i,k} m_{k,n} \quad (\text{Replace Sum})$$

$$m_{i,n}^{(2)} = 1 + \sum_{k=i}^{n-1} p_{i,k} m_{k,n}^{(2)} + 2 \sum_{k=i}^{n-1} p_{i,k} m_{k,n} \quad (j < i \Rightarrow p_{i,j} = 0, i \neq n)$$

$$m_{i,n}^{(2)} = 1 + p_{i,i} m_{i,n}^{(2)} + \sum_{k=i+1}^{n-1} p_{i,k} m_{k,n}^{(2)} + 2 \sum_{k=i}^{n-1} p_{i,k} m_{k,n} \quad (\text{Expand sum})$$

$$m_{i,n}^{(2)}(1-p_{i,i}) = 1 + \sum_{k=i+1}^{n-1} p_{i,k} m_{k,n}^{(2)} + 2 \sum_{k=i}^{n-1} p_{i,k} m_{k,n} \quad (\text{Factor, } m_{i,n} \neq 0)$$

$$m_{i,n}^{(2)} = \frac{1 + \sum_{k=i+1}^{n-1} p_{i,k} m_{k,n}^{(2)} + 2 \sum_{k=i}^{n-1} p_{i,k} m_{k,n}}{1 - p_{i,i}} \quad (\text{Divide by } 1 - p_{i,i}, \\ p_{i,i} \neq 1)$$

$$m_{i,n}^{(2)} - m_{i,n}^2 = \frac{1 + \sum_{k=i+1}^{n-1} p_{i,k} m_{k,n}^{(2)} + 2 \sum_{k=i}^{n-1} p_{i,k} m_{k,n}}{1 - p_{i,i}} - m_{i,n}^2 \quad (\text{Subtract } m_{i,n}^2 \text{ from} \\ \text{both sides})$$

$$v_{i,n} = \frac{1 + \sum_{k=i+1}^{n-1} p_{i,k} m_{k,n}^{(2)} + 2 \sum_{k=i}^{n-1} p_{i,k} m_{k,n}}{1 - p_{i,i}} - m_{i,n}^2 \quad (\text{Replace LHS by } v_{i,n})$$

□

Theorem 2.2 provides a direct computation of $v_{i,n}$, when $i \neq n$, which takes advantage of the fact that P is nearly upper triangular. Recall that when $i = n$, there is one entry below the main diagonal of the single-step transition matrix, namely $p_{n,1}$. Thus, when

$$\begin{aligned} v_{n,n} &= \sum_{k \neq n} p_{n,k} m_{k,n}^{(2)} + 2 \sum_{k \neq n} p_{n,k} m_{k,n} + 1 - m_{n,n}^2 \\ &= (p_{n,1} m_{1,n}^{(2)} + 2 p_{n,1} m_{1,n} + 1 - m_{n,n}^2). \end{aligned}$$

$i = n,$

Chapter 3

Question 1 - Determine Field Reliability

The first question of SRG concerning the stage-wise Binomial failures, one-at-a-time removable, is as follows: “After a given number of system tests, what is the (approximate) probability that the system will operate satisfactorily (not fail) when released to the field or delivered to a user?”

3.1 Computing the Field Reliability

Two quantities are needed to answer this question. The first quantity, Q_i , is the stage-wise survival probability vector of the system as fielded, i.e., the probability a remaining defect in Stage i does not activate while the system is put in use during one mission. The second quantity is the joint probability of the number of defects present in each stage after t tests, denoted $p(d_1, \dots, d_s, t)$.

3.1.1 Stage-Wise Survival Probabilities of the Fielded System

Denote the probability that a remaining defect in Stage i does not activate while the system is put in use in the field during one mission by Q_i , $1 \leq i \leq s$. Then, the probability

that no defect is activated while the system is put in use is $Q = \prod_{i=1}^s Q_i^{d_i}$. It is desirable for

the Q_i to be greater than or equal to the probability that a test does not reveal a defect in Stage i .

Use the binomial defect survival probabilities of the system under test as a lower bound for the survival probabilities of the fielded system. Thus, we have $Q_i \geq \theta_i$ and

$Q_i^{d_i} \geq \theta_i^{d_i} = \tilde{q}_i(d_i)$, $1 \leq i \leq s$. As noted in section 2.1.2, the probability of a successful end-

to-end test of the system is given by $\prod_{i=1}^s \tilde{q}_i(d_i)$.

Since state j loops back to itself on a successful test of the system (see Figure 2.1), these are the $p_{j,j}$ entries of the transition matrix. Thus, the product of the stage-wise survival probabilities for state j is bounded below by $p_{j,j}$. Given a specific number of defects remaining in each stage after t tests, a lower bound for the probability of survival of the

fielded system is given by $Q(j) = \prod_{i=1}^s Q_i^{d_i} \geq \prod_{i=1}^s \tilde{q}_i(d_i) = p_{j,j}$, where j denotes the state

of the Markov chain testing model representing the specified number of defects in each of

the s stages. For the purpose of testing, we shall assume that these two probabilities are equal.

3.1.2 Joint Probability of Defects Remaining in Each Stage After t Tests

In SRG, the joint probability of the number of defects remaining after t tests is defined as

$$p(d_1, \dots, d_s, t) = \Pr[D_1(t) = d_1, \dots, D_s(t) = d_s].$$

In the following Markov chain the joint probability of the number of defects remaining after $t + 1$ tests is determined by summing the probability of no defects being removed on test t and the probability of one defect being removed on test t ,

$$\begin{aligned} p(d_1, \dots, d_s, t) &= p(d_1, \dots, d_s, t) \prod_{j=1}^s \tilde{q}_j(d_j) \\ &+ \sum_{i=1}^s p(d_1, \dots, d_i + 1, \dots, d_s, t) \left(\prod_{j=1}^{i-1} \tilde{q}_j(d_j) \right) \tilde{p}_i(d_i + 1) \end{aligned}$$

These probabilities may be computed from P as follows.

The Markov chain testing model is designed such that each state represents the system with a unique distribution of defects among the s stages. The joint probability of the number of defects remaining after t tests may be interpreted as the probability of being in state i , $1 \leq i \leq n$, after t tests, given that the process started in the model source. Since we start testing from the source, take the initial probability vector, π_0 , to be the $1 \times n$ row vector

with 1 in the first component and 0 elsewhere. Now, $\pi_t = \pi_0 P^t$, but $\pi_0 P^t$ is the first row of the matrix P^t . Thus, the first row of the t -th power of the single-step transition matrix gives the probability of being in each of the different states after t tests, given that the process started at the source [3].

3.1.3 Probability of System Survival in the Field After t Tests

Denote the probability of system survival in the field after t tests as $\tilde{Q}(t)$. In SRG, this probability is determined by taking the sum over all possible distributions of defects of the joint probability of the number of defects present in each stage after t tests multiplied by the probability that no defects are activated during execution of the system, i.e.,

$$\tilde{Q}(t) = \sum_{d_1, \dots, d_s} p(d_1, \dots, d_s, t) \prod_{j=1}^s Q_j^{d_j}.$$

It will be useful to construct a $n \times 1$ column vector, β , whose j th component is $P_{j,j}$. Theorem 3.1 presents an equivalent solution using the initial probability vector, the t th power of the transition matrix, and the β vector.

Theorem 3.1 The probability of system survival in the field after t tests is given by

$$\tilde{Q}(t) = \pi_0 P^t \beta.$$

Proof

$$\tilde{Q}(t) = \sum_{d_1, \dots, d_s} p(d_1, \dots, d_s, t) \prod_{j=1}^S Q_j^{d_j} \quad (\text{Given})$$

$$\tilde{Q}(t) = \sum_{d_1, \dots, d_s} p(d_1, \dots, d_s, t) \prod_{j=1}^S \tilde{q}_j(d_j) \quad (Q_j^{d_j} = \tilde{q}_j(d_j))$$

$$\tilde{Q}(t) = \sum_{i=1}^n p_{1,i}^{(t)} p_{i,i} \quad \left(\sum_{d_1, \dots, d_s} p(d_1, \dots, d_s, t) = \sum_{i=1}^n p_{1,i}^{(t)}, \right. \\ \left. q_j(d_j) = p_{i,i} \right)$$

$$\tilde{Q}(t) = \pi_0 P^t \beta$$

□

Given the number of stages in the system, the number of defects initially present in each stage, and the defect survival probabilities, this model may be automatically generated and the desired results obtained by Algorithms 1 and 2 of the Appendix, respectively

3.2 Larger Question and Answer Space

Since the initial probability vector and transition matrix determine the entire Markov process, we are able to answer additional questions concerning the testing process. These questions and their corresponding answers may be considered in four separate categories: those which concern the model (and therefore the testing process as a whole), those which

deal with the states of the model, those which deal with the stimuli, and those which deal with the arcs of the model.

3.2.1 Further Questions Concerning the Testing Model

One question of interest is the expected number of tests to run until all defects are removed from the system. Since at most one defect is identified and removed per test, and no additional defects are introduced, it is clear that a testing strategy following current assumptions will tend toward total defect elimination. Indeed, the sink of the testing model, which represents testing the system with zero defects in each stage, is an absorbing state. It also comprises the only ergodic set of states in the model.

Since each state transition in the Markov chain testing model represents one test of the system, the expected test case length may be interpreted as the expected number of tests to run before all defects have been identified and removed from the system under test. Similarly, the variance of the expected test case length may be interpreted as the variance of the number of tests to run before all defects have been removed from the system.

3.2.2 Further Questions Concerning the States of the Testing Model

From Gaver's results, it is known that the values of the defect survival probabilities play a key role in the outcome of the testing process. There are many analytical results which provide relevant information concerning the testing process as modeled. Moreover, since these results may be obtained for multiple probability distributions of a particular

model, comparisons may be made between the different probability distributions, providing further information upon which to design a testing strategy. Results of particular interest are the probability of occurrence of a state, the mean occurrence and variance of a state, as well as the mean first passage times and their variances for the individual states.

How many times will a particular state, and thus a particular distribution of design defects, be visited as testing progresses, and what is the associated variance? This question may be answered by calculating the mean occurrence and associated variance of the states in the testing model. The mean occurrence of a state within the Markov process may be used to evaluate and revise the defect activation and survival probabilities within the stages, allowing testers to weight the model. Proper weighting may then be used to ensure more testing of desired stages as in SRG.

Given some number of design defects within each stage of the system, how many tests are expected to be run before a certain distribution of design defects is reached? The $m_{i,j}$ entries of M give the mean number of tests to run before reaching state j , given that the process is in state i . Since each state of the testing model represents a unique distribution of design defects, test managers need only determine the indices of the two states, and check that entry in M .

3.2.3 Further Questions Concerning the Stimuli of the Testing Model

The arcs of the testing model are labeled either “success” or “failed_stage i ”, indicating a successful test or a failure in Stage i , respectively. Thus, statistical information about the stimuli of the model provides information about the survival and failure characteristics of

each stage. Additional results of interest include the long run occupancy of each stimulus, and the mean occurrence and variance of each stimulus.

The long-run occupancy of the stimuli gives the amount of time that the testing process either fails in a given stage, or succeeds. Since the stimuli of the testing model represent either a successful test of the system or a failure in a particular stage, this information may be interpreted as failure characteristics of the various stages as the testing process unfolds. The state transition probabilities of the testing model may be modified to more accurately reflect the behavior of the testing process if the long run occupancies of the stimuli do not agree with the anticipated behavior of the testing process.

The mean occurrence of a stimulus in the testing process is interpreted as the expected number of times that a particular stimulus will occur. The mean occurrence may be used by testers to determine if more testing is needed to identify and remove all defects from a particular stage, and also whether or not the testing model behaves as intended.

3.2.4 Further Questions Concerning the Arcs of the Testing Model

What is the probability that a defect will become active in a particular stage given a specified distribution of design defects among the stages of the system? The answer to this question is provided by the probability of occurrence of the arc representing a failure in the stage in question for the specified distribution of defects. The mean occurrence of an arc in the testing process provides further information for the testers to use in setting the defect activation and survival probabilities for the stages.

3.3 An Example of System Reliability Growth

Section 4.1 of SRG presents an example of the probability of system survival in the field after t tests. The system modeled in the example consists of four stages, with three defects initially present in each stage. The defect survival probabilities are given in Table 3.1. The experiment was conducted using the Markov chain testing model presented in Chapter 2. A graph of the results obtained from the Markov chain testing model is given in Figure 3.1. Visual inspection suggests that the results obtained from the Markov chain testing model accurately reproduce the results obtained by Gaver in SRG.

An analysis of the testing model was performed using the default distribution of Table 3.1. Due to the size of the model (256 states and 1023 arcs), only an excerpt of the analysis is provided. The analysis shows that the expected test case length is 13.606 events. This may be interpreted as the expected number of tests to run before all defects have been identified and removed from the system. The variance associated with the expected test case length is 2.744 events, or tests.

Suppose that sometime during the testing process there are two defects remaining in Stage 1, and three defects in all other stages. Statistical analysis of the model shows that the probability of occurrence of state test_2_3_3_3 is 0.876, and that it has a mean occur-

Table 3.1: Defect Survival Probabilities for Example System.

Distribution	θ_1	θ_2	θ_3	θ_4
Default	0.5	0.5	0.5	0.5
Distribution 1	0.25	0.25	0.75	0.75
Distribution 2	0.75	0.75	0.25	0.25

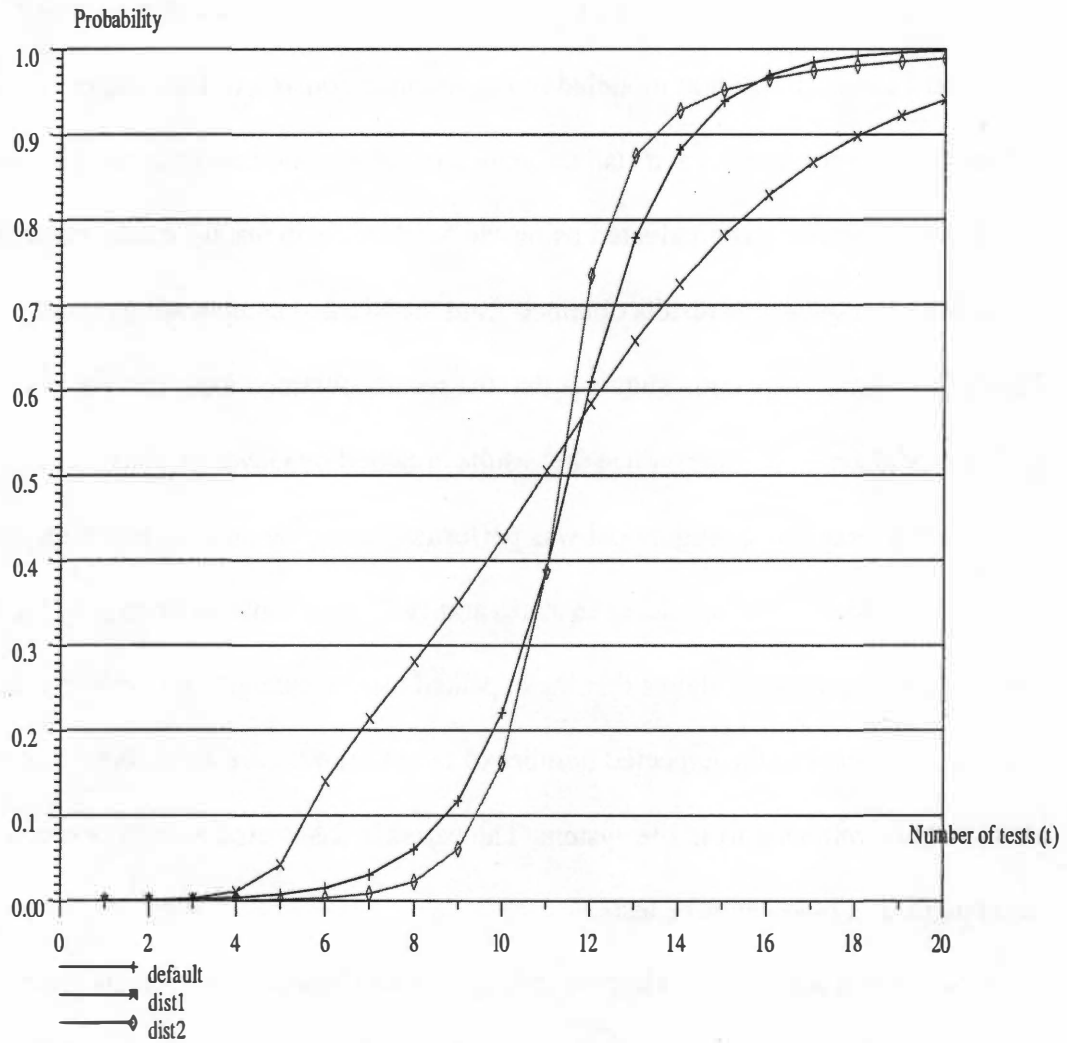


Figure 3.1: Probability of System Survival After t Tests.

rence of 0.875641234 visits per testing scenario, with a variance of 0.109749199 visits. The mean first passage of state test_2_3_3_3 is 3.962 tests, with associated variance of 18.894 tests. The long run occupancy of state test_2_3_3_3 is 59.9489387E-3, meaning that the testing process will be in this state roughly 6 percent of the time.

Table 3.2 gives the long run occupancies and mean occurrences for the stimuli of the model. We see that, on average, the expected failure rates of the stages are equal. This is expected, since default failure probabilities of $\theta = 0.5$ were used for all stages. The mean occurrence of each stimulus representing a failure is 3, as should be expected. The mean occurrence of a success in the testing process is 1.606 tests. Testers may use this data to modify defect activation probabilities of the stages so that the model more accurately models the desired behavior of the testing process. For example, it may be more desirable to have more successful tests run on the system, in which case testers can increase the survival probabilities in stages one and two, while reducing them in the remaining stages.

Table 3.3 presents statistical data about the arcs leading into state test_0_0_0_0. The probabilities of occurrence of these arcs give the probability of entering the model sink from the respective state. The process enters the model sink from state test_1_0_0_0 with a probability of 1.70940171E-3, whereas the probability of entering the sink from test_0_0_0_1 is 0.875213675. These data permit testers to more fully understand the behavior of the process, thereby allowing them to design Markov chain testing models which more accurately represent the testing process.

Table 3.2: Stimulus Statistics for Example System.

Stimulus	Occupancy	Mean Occurrence
fail_stage1	0.220483652	3
fail_stage2	0.220483652	3
fail_stage3	0.220483652	3
fail_stage4	0.220483652	3
success	0.118065394	1.606

Table 3.3: Statistics for Incoming Arcs of State test_0_0_0_0.

From State	Long Run Occupancy	Probability of Occurrence	Mean Occurrence/Variance	
test_1_0_0_0	125.63171E-6	1.70940171E-3	1.70940171E-3	5.12236102E-3
test_0_1_0_0	1.00505368E-3	13.6752137E-3	13.6752137E-3	40.6516181E-3
test_0_0_1_0	8.04042946E-3	0.109401709	0.109401709	0.30426766
test_0_0_0_1	64.3234357E-3	0.875213675	0.875213675	1.094

Chapter 4

Question 2 - Determine the j th Success

The next question addressed by Gaver is, “How many tests are likely to be required to achieve the first (or j th) end-to-end success?” The basic testing model, as presented in Chapter 2, makes a state transition from state i to state i upon a successful test. Modification of the single-step transition matrix so that a successful test makes the transition to the model sink allows the expected number of tests required to achieve the first end-to-end success to be calculated by application of Theorem 2.1. Further modification of the testing model allows the computation of the mean first passage of the model sink to be interpreted as the expected number of tests required to achieve the j th success. The first order Markov property permits these extensions to be made without further modification to the single-step transition matrix.

4.1 The j -Success Model

The states and failure arcs for the j -success model are similar to the testing model as described in Chapter 2. The arcs representing a successful test of the system lead to the

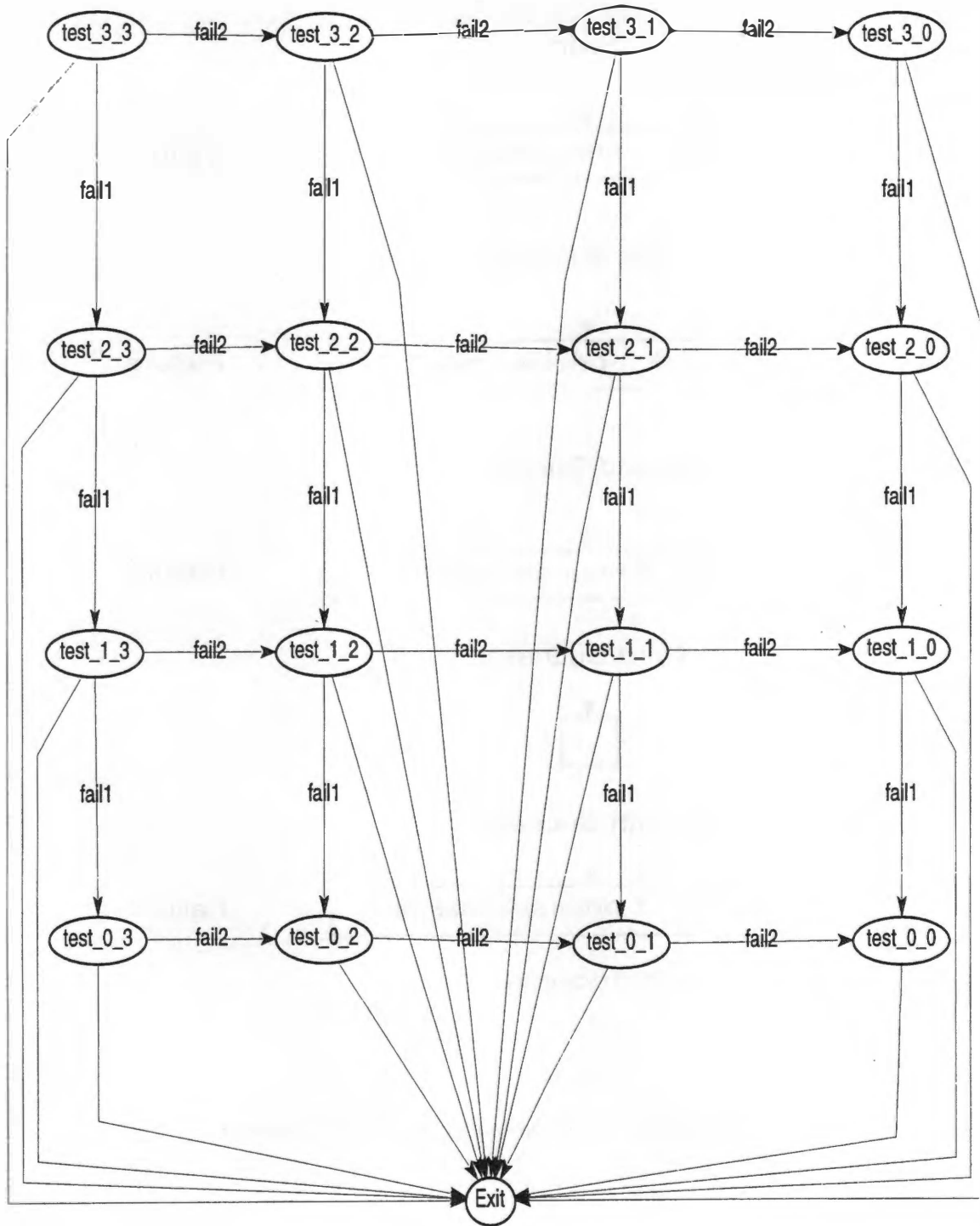
model sink. Figure 4.1 depicts a sample j -success model (for $j = 1$) of a two-stage system, with three defects initially present in each stage. The single step transition matrix for the model may be constructed as follows. Set the diagonal entries of the transition matrix to zero, $P_{i,i} = 0$, and the entries representing a transition from state i to the model sink to

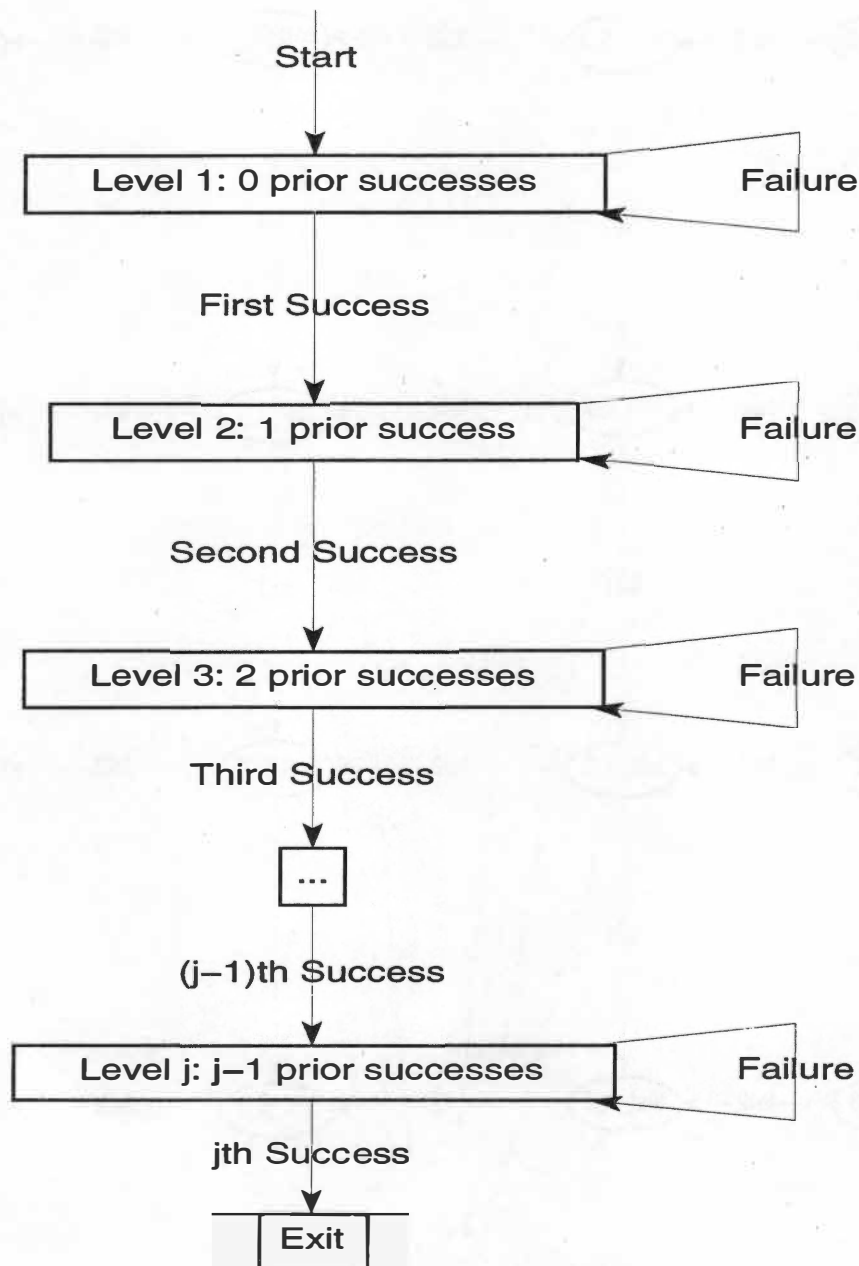
$$P_{i, \text{sink}} = \prod_{l=1}^s \tilde{q}_l(d_l).$$

A successful test then makes the transition to the model sink, allowing the expected number of tests required to achieve the first success to be computed as the mean first passage of the model sink, given the process started in the model source. This model may be further revised (extended) so that the computation of the mean first passage of the model sink gives the expected number of tests to be run before the j th success is obtained.

The model is extended by adding $j - 1$ copies of the 1-success model (Figure 4.1), giving a total of j states representing the same distribution of design defects among the stages.

Label each of these states $\text{test_}d_1 \dots d_s_k$, signifying that the state represents testing the system with $d_1 \dots d_s$ defects and $0 \leq k < j$ prior successes. Figure 4.2 depicts the tiered structure of the j -success model. Each level of the j -success model represents testing the system with some specified number of prior successes. Since a failure in any stage removes exactly one defect from the stage without increasing the number of successes, arcs which represent a failure make the transition to the appropriate state in the same level of the model. Arcs which represent a successful test of the system now make the

Figure 4.1: Example j -Success Model.

**Figure 4.2:** Tiered Structure of j -Success Model.

transition to the next level of the model; a successful test while in state $\text{test_d}_1 \dots \text{d}_s \text{--}k$ makes the transition to state $\text{test_d}_1 \dots \text{d}_s \text{--}k+1$, $0 \leq k < j-2$. A successful test of the system while in state $\text{test_d}_1 \dots \text{d}_s \text{--}j-1$ signifies the j th success, and thus makes the transition to the model sink. Since the model sink may only be reached by a total of j successes, computing the mean first passage of the model sink (from the model source) for the j -success model gives the expected number of tests required to achieve the j th success. An example of a 3-success model for a two-stage system with three defects initially present in each stage is given in Figure 4.3 (rectangles and ellipses distinguish one tier from another). By the first order Markov property, the probability of making a transition is independent of the number of state transitions previously made. Thus, the probability of going from state $\text{test_d}_1 \dots \text{d}_s$ to state $\text{test_d}_1 \dots \text{d}_i \text{--}1 \dots \text{d}_s$, $1 \leq i \leq s$, is independent of the number of prior tests run. Moreover, the probability of going from state $\text{test_d}_1 \dots \text{d}_s \text{--}k$ to state $\text{test_d}_1 \dots \text{d}_s \text{--}k+1$ is independent of the number of prior successes, k . An indexing scheme allows the $n \times n$ single-step transition matrix to be used to obtain the transition probabilities of the j -success model, thereby avoiding the need to store or manipulate the larger $(jn+1) \times (jn+1)$ transition matrix.

Given the number of stages in the system, the number of defects initially present in each stage, and the defect survival probabilities, the j -success model can be automatically generated and the desired results obtained by Algorithms 1 and 3 of the Appendix, respectively.

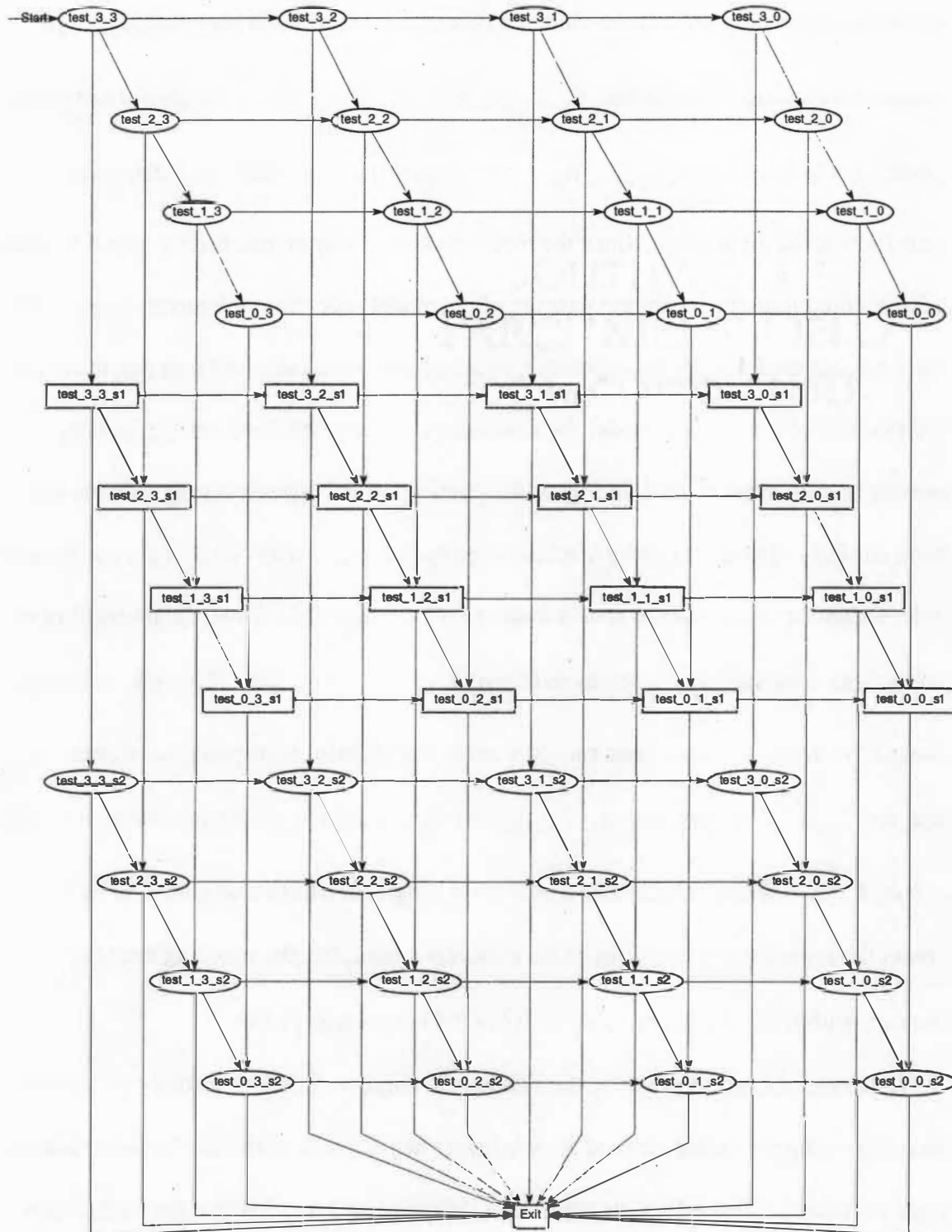


Figure 4.3: 3-Success Model for Two-Stage System with 3 Defects Per Stage.

4.2 Computing the Expected Value of the j th Success

Using the j -success model presented in the previous section, the expected number of tests likely to be run before the j th success may be determined by the mean first passage of the model sink, given that the process started in the model source. There are two quantities required to determine this. The first quantity is the row vector α of long run occupancies of the states. The second quantity is the vector of mean first passage times of the model sink from each state of the model.

4.2.1 Computing the Long Run Occupancies

The long run occupancies of the states may be computed as described in Section 2.2.1. Let $N = jn + 1$, and let y be the $1 \times N$ row vector whose i th component is the probability of occurrence of state i in the j -success model. If the single-step transition matrix for the j -success model were used, then the probability of occurrence for state i could be calculated as $y_i = \prod_{l=1}^i p_{l,i} y_l$. Recall from Section 2.2.1, that each state has at most $s + 1$

incoming arcs. This means that, for large N , P becomes very sparse. Only the nonzero entries of P are used, resulting in a $(s + 1) \times n$ matrix where the $p_{l,i}$ entry, $1 \leq l \leq s$, gives the probability of failing in Stage l while in state i . The $p_{s+1,i}$ entry then gives the probability of success in state i . Working with this compact single-step transition matrix,

Algorithm 3 of the Appendix uses an indexing scheme to obtain the probabilities of occurrence as follows.

The probability of occurrence for the model source is 1, by definition. The probability of occurrence of the model sink is also 1, since the process must eventually reach the model sink. Let k be the number of prior consecutive successes. Let

$\delta_l = \prod_{h=l+1}^s (D_h(0) + 1)$ be the index to the state with the same number of prior success

and one more defect in Stage l , and $\Delta_l = \delta_l + nk$ be the index into the compact P matrix.

Then y_i , $1 < i \leq N-1$, may be computed by $y_i = \sum_{l=1}^{s+1} p_{l,i-\Delta_l} y_{i-\delta_l}$.

4.2.2 Computing the Mean First Passage of the Model Sink

Theorems 2.1 and 2.2 give formulas for the mean first passage times of the model sink (one for each state in the model) and the associated variances. Since the j -success model makes a state transition to the next level of the model upon a successful test, the diagonal entries of the single-step transition matrix are zero. Thus, the mean first passage time of the model sink from state i , $i \neq N$, may be calculated by Theorem 2.1 as

$m_{i,N} = 1 + \sum_{k=i+1}^{N-1} p_{i,k} m_{k,N}$. By the same property, the associated variance may be com-

puted as $v_{i,N} = 1 + \sum_{k=i+1}^{N-1} p_{i,k} m_{k,N}^{(2)} + 2 \sum_{k=i+1}^{N-1} p_{i,k} m_{k,N} - m_{i,N}^2$.

Let the vector of mean first passage times of the model sink be represented (stored) as a $N \times 1$ column vector. The N th state is the model sink. This value is determined by

$$m_{N,N} = \frac{1}{\alpha_N}.$$

The mean first passage of the model sink from the remaining states may be computed using the compact transition matrix. Consider the computation of $m_{i,N}$, where state i represents testing a specific distribution of defects and k prior successes. Let $\gamma = i \pmod{n}$ be the index of the corresponding defect activation in the compact transition matrix. By Theorem 2.1,

$$m_{i,N} = \sum_{l=1}^s p_{l,\gamma} m_{i+\delta_p N} + p_{s+1,\gamma} m_{i+n,N}.$$

4.3 Larger Question and Answer Space

The matrix of mean first passage times provides the mean first passage time of state i given that the process started in state l , $1 \leq i, l \leq N$. Thus, in computing the mean first passage time of the model sink, given that the process started in the model source, the mean first passage times are also calculated for the different levels of the model, i.e., the computation of the mean j th success also calculates the mean i th success, $1 \leq i \leq j$. Moreover, the i th success may be obtained for the system in question for all distributions of design

defects in which the number of defects is less than or equal to the assigned initial number of defects in each stage: $D_h(t) \leq D_h(0)$.

4.3.1 Further Questions Concerning the j -Success Testing Model

The most notable result obtained from the j -success model is the variance associated with the expected value of the j th success. The variance associated with the expected value of a stochastic process can become rather large. Therefore, testing scenarios based solely upon the expected number of tests to run before the j th success may not be as tractable as believed. Given the variance associated with the mean first passage times (one for each state in the model), test planners can develop more realistic testing scenarios which take this variability into account.

4.3.2 Further Questions Concerning the Stimuli of the j -Success Model

The stimuli in the j -success model represent either a successful end-to-end test of the system, or a failure in some stage. Labeling the stimuli to differentiate between successes and failures between the levels of the model allows test planners to understand the failure characteristics of each level, independent of the other levels. This is accomplished by concatenating the number of prior successes to the end of the general stimulus name. Consider, for example, testing a two stage system with three defects present in Stage 1, and two defects present in Stage 2 prior to running the current test. Further suppose that the

testing process has encountered $i - 1$ previous success, and is thus in the i th level of the model. The arc associated with a failure in Stage 2 may be labeled “test_3_2_s($i-1$).

In this manner, test planners can gain a better understanding of the mean number of failures that will occur between successes $i - 1$ and i . Moreover, the Markov property allows the generalization of this result to extend beyond the occurrence of one success and the next, so that the mean number of failures and associated variance between success i and success k are known.

4.4 An Example of Computing the j th Success

Consider testing a system composed of four stages, with three defects initially present in each stage. The state space of the 3-success model is three times larger than the reliability testing model, consisting of 769 states and 3072 arcs. The defect survival probabilities of the four stages were obtained from examples given in SRG and are presented in Table 3.1. The j -success model was used to obtain the expected number of tests to run before the j th success for various values of j . The results are listed in Table 4.1, with the variance included in parentheses. The expected number of tests for each value of j was rounded up to the next integer (since a fraction of a test makes no sense), and applied to Algorithm 2 of the Appendix in order to determine the probability of system survival after the tests were run. A graph of these results is presented in Figure 4.4.

A model analysis was also performed on the j -success model, with $j = 3$. Again, only excerpts of the analysis are presented in illustration. The stimuli of the 3-success model

Table 4.1: Expected Number of Tests Required to Achieve j Successes.

j	Default	Distribution 1	Distribution 2
1	11.396331 (2.7067014)	9.1635279 (3.2660355)	12.030339 (1.3721171)
2	13.393549 (0.7433056)	11.514142 (2.4073126)	13.556935 (0.4798860)
3	14.72672 (0.2994539)	13.288138 (1.7495115)	14.721728 (0.2862185)
4	15.86953 (0.1362793)	14.788527 (1.2545298)	15.80459 (0.1981841)
5	16.936178 (0.0651946)	16.12785 (0.9022459)	16.857148 (0.1441360)
6	17.968428 (0.0319060)	17.364828 (0.6537522)	17.894046 (0.1066280)
7	18.984297 (0.0157854)	18.533675 (0.4772646)	18.920982 (0.0793894)
8	19.992169 (0.0078514)	19.655676 (0.3506295)	19.940933 (0.0592742)
9	20.99609 (0.0039155)	20.74471 (0.2588853)	20.955797 (0.0443195)
10	21.998046 (0.0019552)	21.810156 (0.1918833)	21.966899 (0.0331666)

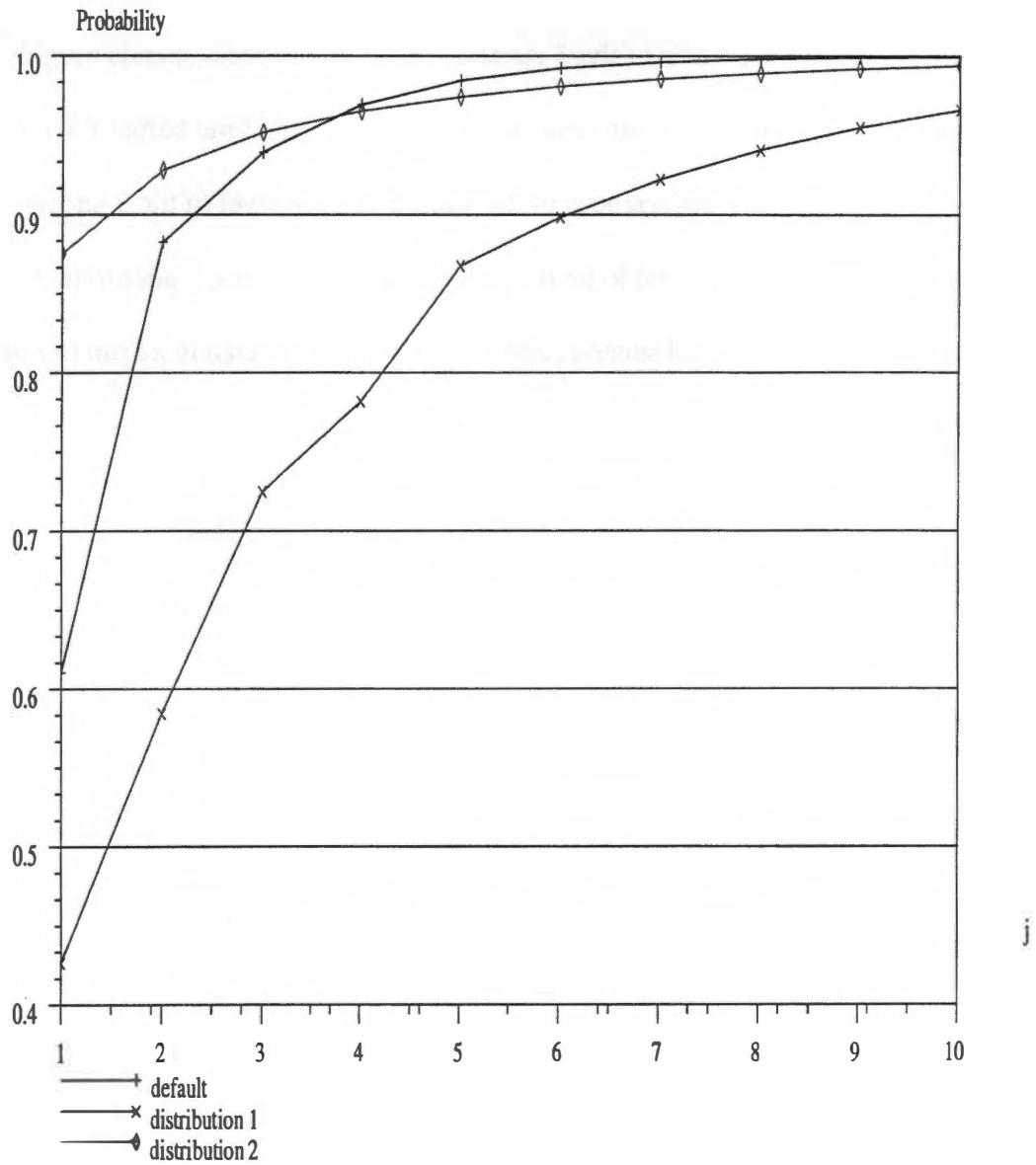


Figure 4.4: Probability of System Survival After j Successes.

were labeled to identify the stimulus with its level in the model. Table 4.2 lists the occupancy and mean occurrence of each stimulus in the testing process. As is seen by summing the occupancies of the stimuli of the 3-success model, the process spends roughly 83 percent of the time before the first success, and 97 percent of the time before the second success. Summing the mean occurrences of the stimuli in each level of the 3-success model shows that 12 tests are expected to be run before the first success, 2 additional tests are expected to obtain the second success, and 1 more test is expected to be run to achieve the third success.

Table 4.2: Additional Results Concerning the Stimuli.

Stimulus	Occupancy	Mean Occurrence
fail_stage1_0	0.2175637	2.985
fail_stage2_0	0.2126591	2.917
fail_stage3_0	0.1920708	2.635
fail_stage4_0	0.1355375	1.859
success_0	0.0728941	1
fail_stage1_1	0.0009948	13.6467471E-3
fail_stage2_1	0.0050603	69.4194632E-3
fail_stage3_1	0.0197961	0.271573357
fail_stage4_1	0.0468402	0.642578125
success_1	0.0728941	1
fail_stage1_2	0.0001813	2.48760698E-3
fail_stage2_2	0.0006289	8.62767517E-3
fail_stage3_2	0.0042433	58.2115974E-3
fail_stage4_2	0.0192648	0.264284328
success_2	0.0728941	1

Chapter 5

Question 3 - The r th Consecutive Successes

The next question addressed in SRG is: “How many tests are required to achieve r (e.g. 3 or 5) consecutive end-to-end test successes, or, in statistical parlance, a (first) run of r ?” The basic model of chapter 2 can again be modified to have r states representing each distribution of design defects, with each representing some prior number of consecutive successes. The states will be ordered differently within the model to accommodate this change in the structure of the model. Theorems 2.1 and 2.2 may be applied to the problem to obtain the expected number of tests needed to achieve the first run of r consecutive successes. The first order Markov property allows the consecutive success model to be constructed without further modification of the single-step transition matrix.

5.1 Consecutive Success Model

The j -success model of chapter 4 was designed to determine the mean number of tests to run before the occurrence of the j th success. The current question imposes the require-

ment that the successes now be consecutive (no failures between successes) and, thus, the consecutive success model is constructed differently than the j -success model.

Begin with the testing model as described in Chapter 2 as the base model. Construct $r - 1$ additional states for each state in the original model. These new states represent the same distribution of design defects as in the original, or parent, state, with some specified number k , $0 < k < r$, of prior consecutive successes. The parent states represent no prior successes. The arcs which represent a successful test of the system make the transition to the state representing the same distribution of design defects, but one more consecutive success. The arc representing a successful test of the system given $r - 1$ consecutive successes makes the transition to the model sink.

Arcs representing a failure in Stage i make the state transition to the state with one less defect in Stage i . Since the successes must be consecutive, defect activation in any stage forces the process to make a transition to a state with no prior successes. For example, activation of a design defect in Stage i while testing state $\text{test_d}_1 \dots \text{d}_i \dots \text{d}_s \text{_}k$, $1 \leq i \leq s$, makes the transition to state $\text{test_d}_1 \dots \text{d}_{i-1} \dots \text{d}_s \text{_}0$. In this way, only a sequence of r consecutive successes is capable of reaching the model sink. Figure 5.1 shows the consecutive success model (with $r = 3$) for a system composed of two stages with one defect initially present in each stage. Notice that a successful test makes the transition to the state with one more consecutive success (arcs directed down the model), and that a failure in any stage makes the transition to the respective state with no

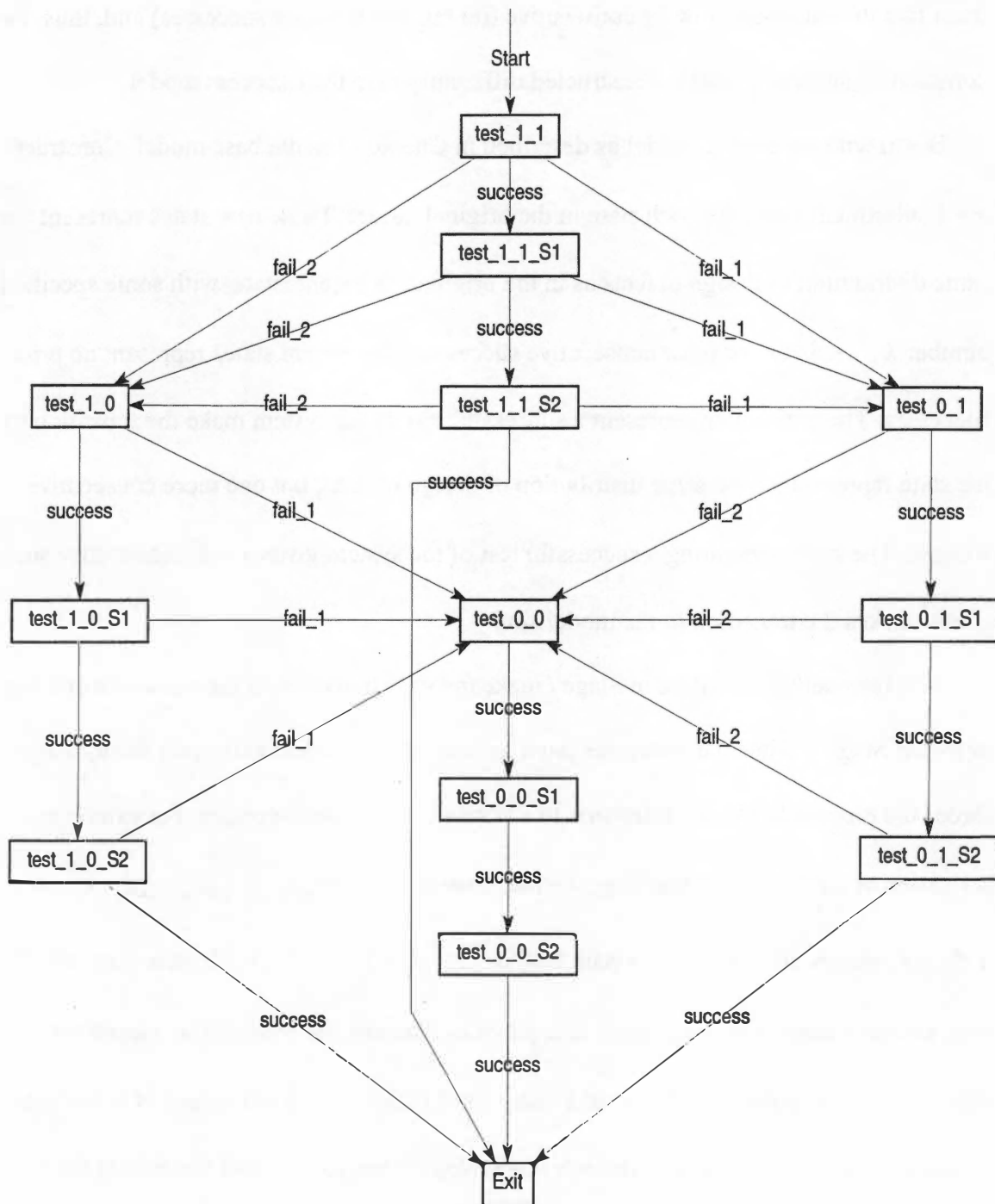


Figure 5.1: Example State Transition Diagram of the Consecutive Success Model.

prior successes (arcs directed across the model). The structure of the example model shows that the sink may only be reached by a series of 3 consecutive successes.

The states may be ordered to maintain the near-upper triangular form of the single-step transition matrix. Consider a system comprised of four stages, with three defects initially present in each stage. Let the state $\text{test_}d_1 \dots d_4_k$ represent testing the system with d_i defects remaining in Stage i , and k prior successes. All states with the same distribution of defects are grouped together, and sorted in increasing order by the number of prior successes. These sorted sets of states are then placed in decreasing order of the number of defects within each stage. Table 5.1 gives the ordering for the above example. This ordering maintains the near-upper triangular form of the transition matrix without the creation of recurrent arcs in the model. Thus, the number of tests likely to be needed before the first run of r consecutive end-to-end tests of the system is given by the mean first passage of the model sink of the consecutive success model, given that the process begins in the model

Table 5.1: Ordering of Sates in the Consecutive Success Model.

State Index	State Name
1	test_3_3_3_3_S0
2	test_3_3_3_3_S1
3	test_3_3_3_3_S2
4	test_3_3_3_2_S0
5	test_3_3_3_2_S1
6	test_3_3_3_2_S2
7	test_3_3_3_1_S0

source. This allows the expected number of tests before the first run of r consecutive successes to be calculated by application of Theorem 2.1. The variance associated with the mean first passage may be computed by Theorem 2.2.

5.2 The Expected Value of the r th Consecutive Success

In SRG a function $r_r(d_1, d_2, \dots, d_s)$ is presented as a solution to determine the expected number of tests to be run until a run of r successes first occurs, given that there are initially d_i defects in Stage i . The consecutive success model may be used to obtain this result. Since the model sink may only be reached by obtaining r consecutive successes during the testing process, the mean first passage of the model sink yields the expected number of tests before the first run of r consecutive successes is achieved.

5.2.1 Computing Probabilities of Occurrence and Long Run Occupancies

The above ordering of the states (Section 5.1) within the testing model simplifies the computation of the probability of occurrence of a state. The constraint that the successes be consecutive means that a state representing a particular distribution of design defects and k prior successes may only be reached from the state with the same distribution of defects and $k - 1$ prior successes. Let y be the $r \times n$ matrix whose (k, i) th element is the probability of occurrence of the distribution of design defects represented by state i with k prior successes in the testing model. Thus, for $k > 0$, $y_{k, i} = y_{k-1, i} p_{s+1, i}$. This is simply

the probability of occurrence of the state representing the distribution of defects to stages, and one less consecutive success multiplied by the probability of a successful test. Algorithm 4 of the Appendix uses an indexing scheme to obtain the probabilities of occurrence from the compact transition matrix.

5.2.2 Computing the Mean First Passage of the Model Sink

Since the consecutive success model makes a state transition to the state representing one more consecutive success upon a successful test, the diagonal entries of the single-step transition matrix are zero. Theorems 2.1 and 2.2 are applied as in Section 4.2.2.

Let the vector of mean first passage times of the model sink (one for each state in the model) be represented (stored) as a $r \times n$ matrix M . Consider the computation of $m_{k,i}$, where state i represents testing a specific distribution of defects with $k < r$ prior successes.

Let $\Delta_j = \prod_{h=j+1}^s (D_h(0) + 1)$ be the index of the state with one less defect in Stage i .

Then by Theorem 2.1, $m_{k,i} = 1 + \sum_{j=1}^s P_{j,i} m_{1,i+\Delta_j} + P_{s+1,i} m_{k+1,i}$.

5.3 Larger Question and Answer Space

Application of the consecutive success model to the problem provides further results which capture the behavior of the testing process. The variance associated with the

expected number of tests to obtain the first run of r consecutive successes can be computed, and serves the same uses as in the previous chapter. The consecutive success model also provides further information about the behavior of the process within each distribution of design defects, as well as success and failure information given a specified number of prior successes.

5.3.1 Further Questions About States of the Consecutive Success Model

The probability of occurrence of a state in the testing process is calculated as a preliminary result when computing the mean first passage of the model sink. Since each state represents a unique distribution of defects among stages, this data can be used to determine the probability of achieving the first run of r consecutive successes from any distribution of defects. Testers may use this information to check the structure of the model,

5.3.2 Further Questions About Stimuli of the Consecutive Success Model

The stimuli of the consecutive success model are named so that failures and successes are mapped to the number of consecutive success. For example, the stimulus representing a failure in Stage 1 may be labeled `fail_Stage1_k`, representing k prior consecutive successes, with $0 \leq k < r$. Statistical analysis of the stimuli may then be used to determine the number of times that k consecutive success are obtained prior to achieving the first run of r consecutive successes using the mean occurrence of an arc in the testing process.

5.4 An Example of the r th Consecutive Success

This section presents an example of computing the expected number of tests to run before the first run of r consecutive successes occurs. The example system is composed of four stages, with three defects initially present in each stage. The example was modeled with three different distributions of defect survival probabilities, obtained from SRG. The defect survival probabilities are given in Table 3.1. A model analysis was performed on the consecutive success model (769 states and 3072 arcs) with $r = 3$.

Algorithm 4 of the Appendix was used to determine the expected number of tests to run before the r th consecutive success, $1 \leq r < 10$. The expected number of tests for each probability distribution and each value of r may be seen in Table 5.2. The expected number of tests was rounded up to the next integer value, since there are no fractional tests applied. These results were then given as input to Algorithm 2 of the Appendix to obtain the probability of system survival after testing. These results are shown in Figure 5.2.

The analysis of the consecutive success model shows that the state test_1_2_3_3 will occur during the testing process with probability 0.50330112, and that the expected number of tests required to reach state test_1_2_3_3 is 19.657. However, the analysis also shows that 16 tests are expected to be run to achieve the third success. Testers may use the analysis of this and other states to modify the defect activation probabilities so that testing reveals more failures in the later stages.

Table 5.2: Expected Number of Tests Required to Achieve r Consecutive Successes.

r	Default	Distribution 1	Distribution 2
1	11.396331	9.1635279	12.030339
2	14.220291	13.081757	14.004584
3	15.87683	16.003546	15.405692
4	17.202149	18.282784	16.675941
5	18.379484	20.16911	17.890302
6	19.479163	21.810475	19.068331
7	20.535507	23.29013	20.217097
8	21.567215	24.657026	21.340804
9	22.58492	25.941884	22.442958
10	23.59472	27.165191	23.526749

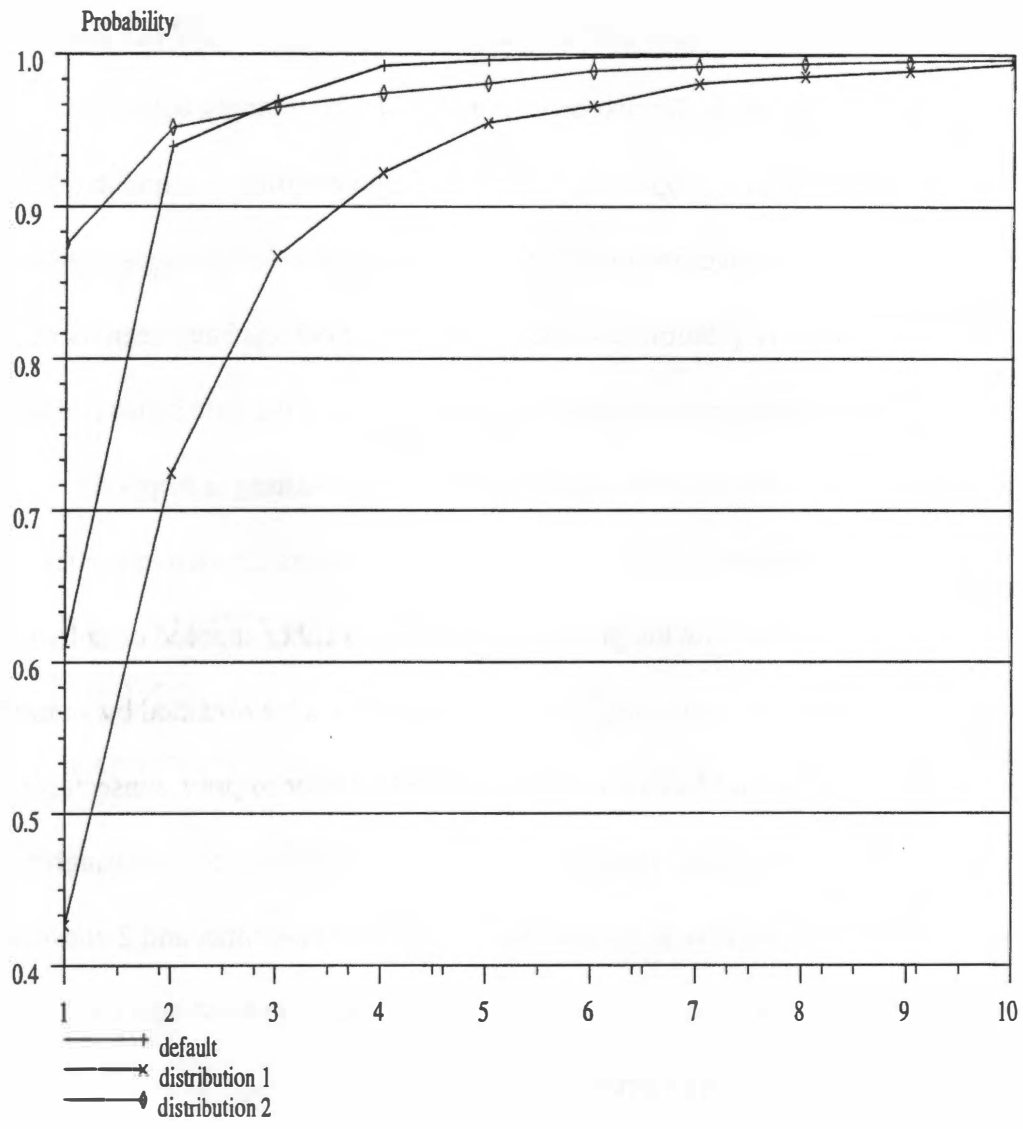


Figure 5.2: Probability of System Success after r Consecutive Successes.

Additionally, the state test_1_2_3_3_S1 , representing one consecutive success so far, has a probability of occurrence of $983.009999\text{E-}6$, and test_1_2_3_3_S2 has probability of occurrence of $1.9199414\text{E-}6$. The expected number of tests to apply before the first run of r can be used as a criteria to stop testing. Thus, if the probability of achieving the r th consecutive success with a specified distribution of defects among the stages is greater than desired, testing may stop before a sufficient number of defects have been identified and removed. This information can be used by testers to adjust the defect survival probabilities of the stages so that more defects are removed before testing is stopped.

The stimuli of the consecutive success model were labeled as detailed in section 5.3.2. Table 5.3 lists the amount of time the process is expected to either succeed or fail with the given number of consecutive successes. The failure statistics were obtained by summing the long run occupancies of all failure arcs for the given number of prior consecutive successes. We see that the process is expected to have no consecutive successes roughly 81 percent of the time, 1 consecutive success about 12 percent of the time, and 2 consecutive successes roughly 7 percent of the time. As a whole, the testing process succeeds 25 percent of the time and fails roughly 75 percent of the time.

We can sum the mean occurrence for each stimulus representing a given number of consecutive successes to obtain the mean number of tests expected to be run with the given number of consecutive successes, as listed in Table 5.4. As the data shows, nearly 13 tests are expected to be run with no consecutive success, 1.8 tests are expected to be run with 1 success, and 1.2 tests are expected to be run with two consecutive success.

Table 5.3: Success and Failure Data for $r = 3$.

Consecutive Successes	Occupancies		
	Success	Failure	Total
0	0.1161066	0.692748	0.8088546
1	0.0745439	0.0415627	0.1161066
2	0.0627297	0.0118143	0.074544

Table 5.4: Mean Number of Tests Applied with Given Consecutive Successes.

Consecutive Successes	Mean Number of Tests
0	12.793
1	1.8363819
2	1.1790351

Chapter 6

Question 4 - Determine Failure Characteristics

The last operationally relevant questions addressed by Gaver is: “Suppose testing is stopped after T tests, after which no further design modifications are contemplated. What are the failure characteristics of the system if fielded: e.g. what is the operational/field probability of system (reliability) success? What is the probability that the system completes a mission that requires M successes if $M + R$ systems are allocated? What is the mean, and variability, of the number of tests required?”

Algorithm 3 or 4 may be used to determine T , the expected number of tests after which testing is stopped, as well as the associated variance. T may then be given as input to Algorithm 2 of the Appendix to determine the operational probability of system success. The probability that the system completes a mission that requires M successes if $M + R$ systems are allocated can be computed using the Binomial distribution.

The number of tests to run is dependent on the stopping criteria used. For example, suppose that the system under test is composed of four stages, with three defects initially

present in each stage, and that testing stops after 5 successes have been encountered.

Using Algorithm 3 of the Appendix, we find that the expected number of tests to obtain 5 successes is $T = 16.936178$ tests, with a variance of 0.0651956 tests. Using T as input to Algorithm 2 of the Appendix, we find that the operational reliability of the system is 0.9845003, roughly 98 percent.

Using the Binomial distribution, the probability that the system completes a mission that requires M success if $M + R$ systems are allocated is computed as

$$p(\text{successes} \geq M) = \sum_{i=M}^{M+R} \binom{M+R}{i} 0.9845003^i (1 - 0.9845003)^{M+R-i}. \text{ Suppose that a}$$

particular mission requires 10 successes, i.e. $M = 10$. Table 6.1 lists the probabilities of mission success for various values of R . We can see that the probability of mission success is 0.9999999 when 15 systems are allocated.

Table 6.1: Probability of Mission Success.

R	Probability of Mission Success
0	.9845003
1	.9879601
2	.9992624
3	.9999631
4	.9999984
5	.9999999

Chapter 7

Summary and Conclusion

Each of the original questions of SRG was answered by designing a special Markov chain and then casting the question as a somewhat conventional statistic of the Markov chain. Each Markov chain has a regular structure which made it possible to find an efficient algorithm to compute the statistics. This approach gave insight to the question and answer space surrounding each of the four original questions of SRG.

The regular structure of each of the proposed Markov chain models allows the computation of results to be done efficiently. It was possible to avoid the traditional matrix solution approaches, resulting in a computational time of $O(n \log n)$ for the worst case.

The Markov chain testing model is constructed from three items of information, namely, the number of stages in the sequential stage system, the number of defects in each stage prior to testing, and the probability that a defect in a stage does not become active. The number of states in the model is determined by the number of stages and the number of defects within each stage. The stage-wise defect activation and survival probabilities for each stage are determined by the number of defects in the stage and the probability that a defect does not become active. Given this information, Algorithm 1 of the Appendix will

generate the single-step transition matrix in compact form, storing only the non-zero entries of the sparse matrix.

The number of stages and the number of defects within each stage also serve as a useful means of indexing when working with the models of Chapters 4 and 5. The ordering of the states permits the desired index to be computed by a simple product of defects within stages. It also simplifies implementation of matrix multiplication in the solution to the probability of system survival after t tests. Thus, all models and all results may be automatically generated.

Follow-on research may generalize current results to extend the method beyond sequential-stage systems, allowing for implementation of a more general statistical model for defect activation and removal.

From the point of view of the software testing experience of SQRL, the larger question and answer space is definitely more versatile and interesting. We will be interested to learn from Gaver, et. al., whether or not value was added to their testing problem.

REFERENCES

References

- [1] D.P. Gaver, P.A. Jacobs, K.D. Glazebrook, E.A. Seglie, "Probability Models for Sequential-stage Reliability Growth *via* Failure Mode Removal," Naval Post-graduate School, 2000.
- [2] J.H. Poore and C.J. Trammell, "Application of Statistical Science to Testing and Evaluating Software Intensive Systems," *Statistics, Testing, and Defense Acquisition*, National Academy Press, Washington D.C., 1998.
- [3] J.G. Kemeny and J.L. Snell, *Finite Markov Chains*, D. Van Nostrand Company, Inc., 1960.
- [4] S.J. Prowell, "Computations for Markov Chain Usage Models," *University of Tennessee Department of Computer Science Technical Report*, 2002.

APPENDIX

Appendix

This appendix presents the algorithms used in the computations. The algorithms were coded to run in Scilab, a free scientific software package for numerical computations available from <http://www-rocq.inria.fr/scilab> (while similar to MATLAB, Scilab programming uses a slightly different syntax).

A.1 Algorithm 1 - The Single-Step Transition Matrix

Algorithm 1 builds the compact $(s + 1) \times n$ single step transition matrix in $O(n)$ time. Inputs are the vector of stage-wise defect survival probabilities, and the vector of the number of defects in each stage.

```
function [P,n,] = build_matrix(theta, defects),

// Get the number of stages in the SUT.
[nr,nc]=size(theta);
stages = nc;
index = 1;

// Get the number of states in the model, and set up the
// counter vector of defects remaining.
n = 1;
for i = 1:stages
    counters(1,i) = defects(1,i);
    n = n*(defects(1,i)+1);
end
```

```

// Add one for the model sink.
n = n+1;

// Create the transition matrix of proper dimensions.
P = zeros(stages+1,n);

// Now, compute the state transition probabilities.
for i = 1:n-1
    survive = 1;
    for j = 1:stages
        P(j,i) = survive * (1 - theta(1,j)^counters(1,j));
        survive = survive * theta(1,j)^counters(1,j);
    end
    P(stages+1,i) = survive;

    // Decrement the current number of defects in the
    // appropriate stage.
    if counters(1,stages) ~= 0
        counters(1,stages) = counters(1,stages) - 1;
    else
        for c = stages:-1:2
            counters(1,c) = defects(1,c);
            if counters(1,c-1) ~= 0
                counters(1,c-1) = counters(1,c-1) - 1;
                break;
            end
        end
    end
end
end
end

```

A.2 Algorithm 2 - Expected Reliability After t Tests

Algorithm 2 computes the probability of system survival after t tests in $O(n)$ time.

```

function [R,pi_zero]=get_PSSAT(P,theta,defects),
// Get the number of stages in the SUT, and the
// number of tests applied.
[stages,n]=size(P);
stages = stages - 1;
tests = input("How many tests have been applied?");

```

```

// Initialize the probability vector.
pi_zero = zeros(2,n);
pi_zero(2,1)=1;

// Iterate for the desired number of tests. Each pass
// through computes the product of Pi_zero*P, storing
// the result in pi_zero.
for l = 1:tests
    for i = 1:n
        // Success
        pi_zero(1,i)= pi_zero(1,i) + pi_zero(2,i)*P(stages+1,i);
        // Perform the multiplication, summing the results.
        for j=1:stages
            index = 1;
            for k = stages:-1:j+1
                index = index * (defects(1,k)+1);
            end
            // Ensure that the index is not out of bounds, and
            // that the transition probability is non zero.
            if i-index > 0
                pi_zero(1,i) = pi_zero(1,i) + pi_zero(2,i-index)*P(j,i-index);
            end
        end
    end
    // Store the vector in the second row, and zero out the
    // first row for the next pass through the loop.
    pi_zero(2,:)=pi_zero(1,:);
    for i=1:n
        pi_zero(1,i)=0;
    end
end
// The reliability is computed by summing the product of
// each component of the probability vector with the
// probability of a successful test.
R = pi_zero(2,:)*P(stages+1,:)'

```

A.3 Algorithm 3 - Expected Value of the j th Success

Algorithm 3 computes the expected number of tests required to obtain the j th success.

Inputs are the single step transition matrix, the number of states in the matrix, and the vec-

tors of stage-wise survival probabilities and defects per stage. The probabilities of occurrence may be computed in $O(n \log n)$ time. The expectation and variance of the number of tests both require $(s + 1)jn$ multiplications and additions, resulting in a computational time of $O(n \log n)$.

```
function [m,v] = get_PSSAJT(P,n,theta,defects),

//Get the number of stages in the SUT.
[nr,stages]=size(theta);

//Get the number of successes desired.
successes = input("How many successes are desired?");
n = n-1;
nTotal = n*successes + 1;

//Create and initialize the vector of prob. of occurrence.
result = zeros(1,nTotal);
result(1) = 1;

//calculate the probability of occurrence for the initial
//distribution of design defects in each level.
for i=1:successes-1
    result(1,i*n+1)=P(stages+1,1)^i;
end
result(1,nTotal)= 1;

// Now, set the states probabilities of occurrence.
for l = 0:successes-1
    a = l*n;
    for i = 2:n
        survive = 1;
        // Set the probabilities for all failures.
        for j=1:stages
            index = 1;
            for k = stages:-1:j+1
                index = index * (defects(1,k)+1);
            end
            if (i-index) > 0
                result(1,i+a)=result(1,i+a) + result(1,i+a-index)*P(j,i-index);
            end
        end
    end
end
```

```

    if i+a > n & modulo(i,n) ~= 0
        result(1,i+a) = result(1,i+a)+result(1,i+a-n)*P(stages+1,i);
    end
    if modulo(i,n) == 0 & a > 0
        result(1,i+a) = result(1,i+a)+result(1,i+a-n);
    end
end
end

// Sum the probabilities of occurrence for normalization.
normali = 0;
for i = 1:nTotal
    normali = normali + result(1,i);
end

// create and initialize the vectors, and compute the mean
//first passage times for the model sink.
m = ones(1,nTotal);
m2 = ones(1,nTotal);
v = zeros(1,nTotal);
m(1,nTotal) = normali/result(1,nTotal);

for i = nTotal-1:-1:1
    sum1 = 0;
    sum2 = 0;
    // Start with a failure in stage 1, and work through each stage.
    for j = 1:stages
        index = 1;
        // Calculate the index into the matrix for a failure in this stage.
        // This is simply the product of the number of defects in each
        // stage from stage j+1 to stages.
        for k = stages:-1:j+1
            index = index * (defects(1,k)+1);
        end
        // Now, check to ensure that the index into the matrix is
        // less than the number of states, and also that modulo(i,n)
        // is greater than zero (to avoid invalid index).
        if i+ index < nTotal & modulo(i,n) ~= 0
            sum1 = sum1 + P(j,modulo(i,n))*m(1,i+index);
            sum2 = sum2 + P(j,modulo(i,n))*m2(1,i+index);
        // If the index is less than the number of states, and
        // modulo(i,n) == 0, then we have a state with no defects
        // remaining. As such, transition to the next state is
        // certain, so we simply add the mean first passage of the

```

```

// model sink to the sum.
elseif i + n < nTotal & modulo(i,n) == 0
sum1 = sum1 + m(1,i+n);
sum2 = sum2 + m2(1,i+n);
break;
end
end
// Now add in the weight for a successful test.
if i < (nTotal - n - 1) & modulo(i,n) ~= 0
sum1 = sum1 + P(stages+1,modulo(i,n))*m(1,i+n);
sum2 = sum2 + P(stages+1,modulo(i,n))*m2(1,i+n);
end
// Add the sum to the mean first passage, and also
// perform the calculation for the variance.
m(1,i) = sum1;
m2(1,i) = sum2 + 2 * sum1
v(1,i) = m2(1,i) - m(1,i)^2;
end

// Theorem 2.2 is not applicable to the model sink, so
// compute it here.
v(1,nTotal) = m2(1,1) + 2 * m(1,1) + 1 - m(1,nTotal)^2;

```

A.4 Algorithm 4 - Mean of the r th Consecutive Success

Algorithm 4 computes the expectation and associated variance of the number of tests required to achieve r consecutive successes. The number of states in the consecutive success model is the same as in the j -success model. Also, the number of floating point operations is approximately equal in the two models. Thus, Algorithm 4 runs in $O(n \log n)$ time.

```

function [m,v,m2] = cons_success(P,n,defects,theta),
// Get the number of stages in the SUT.
[nr,stages]=size(theta);
// Get the number of successes desired.
successes = input("How many consecutive successes are desired?");

```



```

n = n-1;
nTotal = n*successes + 1;

// Create and initialize the vector of prob. of occurrence.
poc = zeros(successes,n+1);
poc(1,1) = 1;

for i = 2:successes
    poc(i,1)=poc(i-1,1)*P(stages+1,1);
end
poc(1,n+1)= 1;

// Now, set the states probabilities of occurrence.
for i = 2:n
    survive = 1;
    // Set the probabilities for all failures.
    for j=1:stages
        index = 1;
        for k = stages:-1:j+1
            index = index * (defects(1,k)+1);
        end
        if i-index > 0
            sum1 = 0;
            for l=1:successes
                sum1 = sum1 + poc(l,i-index);
            end
            poc(1,i)=poc(1,i) + sum1 * P(j,i-index);
        end
    end
    for j=2:successes
        poc(j,i) = poc(j-1,i)*P(stages+1,i);
    end
end

// Sum the components of the probability of occurrence vector for
// normalization.
normali = 1;
for i = 1:n
    for j = 1:successes
        normali = normali + poc(j,i);
    end
end
end

```

```

// Compute the mean first passage times for the model sink,
// as well as the associated variance.
m = ones(successes,n+1);
m2 = ones(successes,n+1);
v = zeros(successes,n+1);

m(1,n+1) = normali;

for i = n:-1:1
    // Start with a failure in stage 1, and work through each stage.
    for j = 1:stages
        index = 1;
        // Calculate the index into the matrix for a failure in this stage.
        // This is simply the product of the number of defects in each
        // stage from stage j+1 to stages.
        for k = stages:-1:j+1
            index = index * (defects(1,k)+1);
        end

        // Now, check to ensure that the index into the matrix is
        // less than the number of states.
        if i+ index <= n
            sum1 = P(j,i)*m(1,i+index);
            sum2 = P(j,i)*m2(1,i+index);
            for k = successes:-1:1
                m(k,i) = m(k,i) + sum1;
                m2(k,i) = m2(k,i) + sum2;
            end
        end
    end
end

m2(successes,i) = m2(successes,i) + 2*m(successes,i);
// Now, cycle for each success < successes, and
// add the probability of success times the
// mfp for the next state.
for k = successes-1:-1:1
    m(k,i) = m(k,i) + P(stages+1,i)*m(k+1,i);
    m2(k,i) = m2(k,i) + P(stages+1,i)*m2(k+1,i) + 2*m(k,i);
    v(k,i) = m2(k,i) - m(k,i)^2;
end
end

```

Vita

Michael Corum was born in Knoxville, Tennessee on 24 July 1968. He graduated from Doyle High School in June of 1986. He joined the United States Army in August of 1986, where he served as a combat medic until May of 1991. In 1997, he graduated from Pellissippi State Technical Community College with an A.A. in Computer Science. In 2000, Mr. Corum graduated from the University of Tennessee with a B.S. in Computer Science and Mathematics.

Michael Corum was awarded the Master of Science degree in Computer Science from the University of Tennessee in May of 2003.